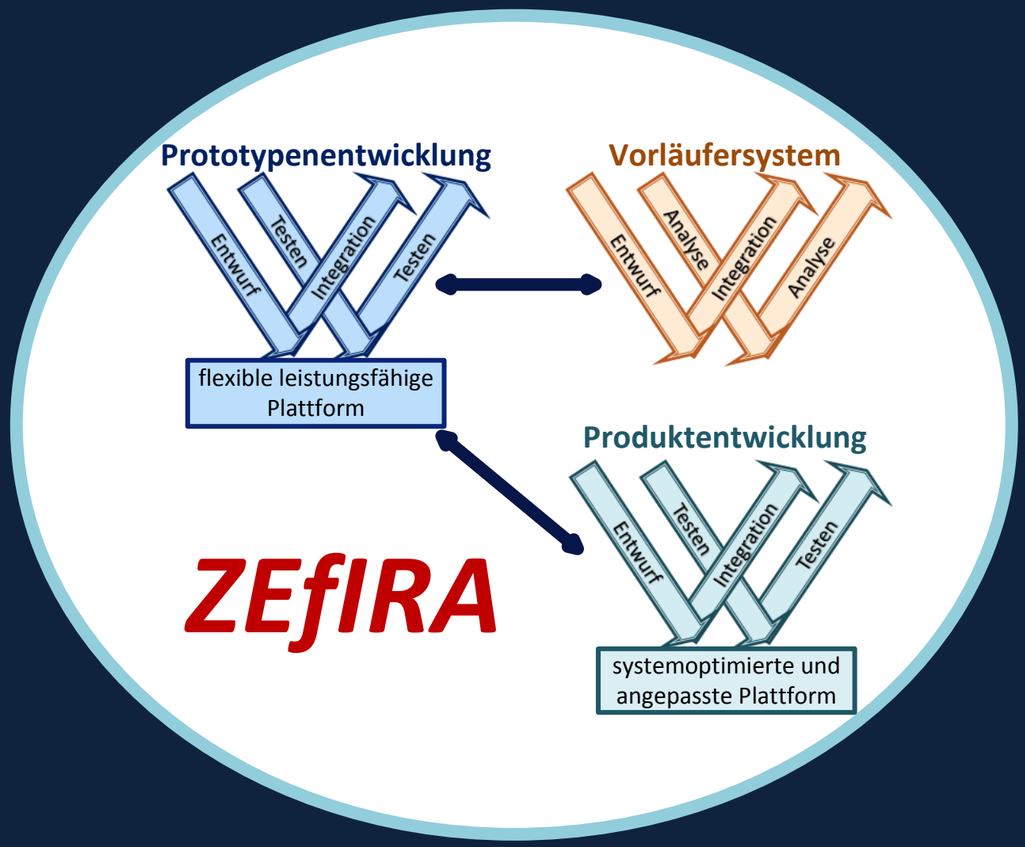
Irina Kaiser

Zertifizierbarer Entwicklungsprozess für komplexe Informationsverarbeitungssysteme in der Wägetechnik

Zertifizierbarer Entwicklungsprozess für komplexe Informationsverarbeitungssysteme in der Wägetechnik

Irina Kaiser



TECHNISCHE UNIVERSITÄT
ILMENAU

Zertifizierbarer Entwicklungsprozess für komplexe Informationsverarbeitungssysteme in der Wägetechnik

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

vorgelegt der

Fakultät für Informatik und Automatisierung

der Technischen Universität Ilmenau

von

MSc. Dipl.-Ing. Irina Kaiser (geb. Gushchina)

geboren am 12. Mai 1986 in Moskau

vorgelegt am: 09.04.2015

Tag der wissenschaftlichen Aussprache: 26.10.2015

Gutachter:
1) Univ.-Prof. Dr.-Ing. habil. Wolfgang Fengler
2) Univ.-Prof. Dr.-Ing. habil. Thomas Fröhlich
3) Prof. Dr.-Ing. Klaus-Dietrich Kramer

urn:nbn:de:gbv:ilm1-2015000588

Kurzfassung

Die vorliegende Dissertation befasst sich mit Prinzipien, Methoden und Techniken der systematischen Entwicklung von komplexen Eingebetteten Systemen, vorwiegend für den Einsatz in der Messtechnik. Die betrachtete Domäne besitzt Anwendungsbereiche mit anspruchsvollen und besonderen Anforderungen an die Informationsverarbeitung. In der dynamischen Wägetechnik sind z.B. Lösungen mit sehr hohen Auflösungen und kleiner Messunsicherheit bei schnellen Messungen in einem mechanisch gestörten Umfeld notwendig. Die Anforderungen an die Eichfähigkeit und die Metrologische Sicherheit sind Besonderheiten dieser Domäne. Es werden komplexe und hochleistungsfähige Funktionen zur Erzeugung der Messergebnisse verlangt, deren Implementierung mit entsprechender Reaktionszeiten, Rechen- und Kommunikationsleistungen realisiert werden muss. In der Arbeit werden dafür vorwiegend FPGA-basierte Eingebettete Systeme verwendet.

Der entworfene zertifizierbare Prozess (ZEfIRA) bietet eine Vorgehensweise für die Entwicklung von Eingebetteten Systemen. Die Metrologische Sicherheit, die Eichfähigkeit, die Validier- und der Verifizierbarkeit werden als Kriterien im gesamten Entwurfsprozess berücksichtigt. ZEfIRA basiert auf einem 3W-Modell und ist evolutionär angelegt. Innerhalb des Prozesses werden die Analyse eines eventuellen Vorläufersystems sowie die modellbasierte prototypische Entwicklung bis hin zu einer produzierbaren Lösung (Produkt) durchgeführt.

Die vorliegende Arbeit verdeutlicht den großen Einfluss der spezifischen Anforderungen an das Messsystem. Es wird gezeigt, wie diese bereits zu der Entwurfszeit auf Modellebene und im Weiteren bei der Implementierung in einer FPGA-basierten Zielplattform berücksichtigt werden. In der Arbeit werden verschiedene Schritte des funktionalen und technischen Systementwurfs untersucht, wobei ausführlich die Realisierungspartitionen „FPGA-Logik“ und „FPGA-Softcore-Lösungen“ betrachtet.

Als Demonstrationsbeispiel zum Nachweis der Anwendbarkeit des Prozesses ZEfIRA inklusive der erarbeiteten Methoden und Prinzipien dient die prototypische Entwicklung des Informationsverarbeitungssystems einer elektromagnetischen Kraftkompensationswaage (EMKW). Ausschlaggebend sind zum einen die optimal an das Gesamtsystem angepassten Signalverarbeitungs-, Regelungs- und Sicherheitsalgorithmen und zum anderen deren technische Umsetzung. Dieses wurde mit verschiedenen Leistungsparametern, wie z.B. Latenz, Verarbeitungskomplexität und Genauigkeit realisiert. Ergänzend ermöglicht der Prototyp umfassende Analysemöglichkeiten für das Einbettende System. Die abschließende Wertung ist eine Abschätzung der Leistungsfähigkeit von ZEfIRA auf Basis dieser prototypischen Entwicklung.

Abstract

The dissertation is about principles, methods and techniques during the systematic development of embedded systems in the domain of measurement techniques. The considered domain contains fields of application with challenging and specific requirements of the information processing system. For example, the dynamic weighing systems need solutions with very high resolution and lowest achievable measurement uncertainty in order to perform high-speed-measurements in a mechanically disturbed environment. In particular, the abilities for official calibration and metrologic reliability are considered. The complex and high-performance functions are required to guarantee measurement results with appropriate computation power, response time and communication performance. FPGA-based embedded systems are used for the implementation of these functions.

The especially designed certifiable development process (ZEfIRA) provides a procedural method for the development of complex embedded systems. The metrologic reliability, the legal requirements like calibratability, the validation and the verification are included as a general criteria in the entire development process. ZEfIRA is based on the 3W-model and is designed in an evolutionary manner. This process starts with the analysis of a predecessor system followed by the model-based development of a prototype, which leads into an optimized and application-specific product solution.

The study emphasizes the influence of challenging requirements on the measurement system. It will be presented, how these can be integrated into the modelling level during the design and the implementation on a FPGA-based target platform. The stages of the functional and technical design of the system are analysed, whereas the realization of the partitions “FPGA logic” and “FPGA softcore solutions” are discussed in detail.

Based on the preliminary design of the information processing in an electromagnetic force compensation (EMC) scale, the applicability of the process ZEfIRA and its developed methods and principles are proved. On the one hand, the optimal system-specific algorithms of signal processing, control and safety and on the other hand whose technical implementation are essential. This was realized with different performance parameters, for example short latency, processing complexity and precision. In addition, the prototype allows the possible comprehensive analysis for embedding system. In the conclusion, the performance of ZEfIRA based on the prototype development is evaluated.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AD	Analog-Digital
ADC (ADU)	Analog-to-Digital-Converter (Analog-Digital-Umsetzer)
AES	Advanced Encryption Standard
ALU	Arithmetic Logic Unit
ASIC	Application-Specific Integrated Circuit
ASIP	Application-Specific Instruction Set Processor
ASM	Assembler
AUTOSAR	AUTomotive Open System Architecture
CDFG	Control/Data Flow Graph
CIM	Computation Independent Model
DA	Digital-Analog
DAC (DAU)	Digital-to-Analog-Converter (Digital-Analog-Umsetzer)
DFG	Deutsche Forschungsgemeinschaft
DIN	Deutsche Institut für Normierung
DLL	Dynamic Link Library
DMA	Direct Memory Access
DMS-WZ	Wägezelle mit Dehnungsmessstreifen
DNL	Differentielle Nichtlinearität
DOORS	Dynamic Object Oriented Requirements System
DSP	Digital Signal Processor
EBS	Einbettendes System
EGS	Eingebettetes System
EMKW	elektromagnetische Kraftkompensationswaage
EU	Europäische Union
FDA	Food and Drug Administration
FDM	Finite-Differenzen-Modelle
FEM	Finite-Elemente-Modelle
FF	Flip-Flop
FFT	Fast Fourier Transform
FG	Fachgebiet
FIFO	First In – First Out
FIR-Filter	Finite Impulse Response Filter
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
HCFSM	Hierarchical Concurrent Finite-State Machine
HDL	Hardware Description Language
HiL	Hardware in the Loop
IIR-Filter	Infinite Impulse Response Filter

IMC	Internal Model Control
INL	Integraler Linearitätsfehler
IP	Intellectual Property
LiCT	LiSARD Code Tester
LiSARD	LabVIEW-integrated Softcore Architecture for Reconfigurable Devices
LSB	Least Signifikant Bit
LUT	Look-up Table
MID	Measurung Instruments Directive
MiL	Model in the Loop
MUX	Multiplexer
MWB	Mittelwertbildner
NaN	Not-a-Number
NI	National Instruments
NPMM	Nanopositionier- und Messmaschine
OIML	International Organisation of Legal Metrology
PC	Personal Computer
PENECA	Petri-Netz-CASE-System
PID-Regler	Proportional–Integral–Derivative Controller
PiL	Processor in the Loop
PIM	Platform Independet Model
PSM	Platform Specific Model
PTB	Physikalisch-Technische Bundesanstalt
PXI	PCI eXtensions for Instrumentation
RAM	Random-Access Memory
RISC	Reduced Instruction Set Computer
ROM	Read-Only Memory
SC	Softcore
SiL	Software in the Loop
SoC	System-on-Chip
SUM	Summierer
SysML	Systems Modeling Language
TINA	TIme petri Net Analyzer
UML	Unified Modeling Language
VHDL	Very High Speed Integrated Circuit Hardware Description Language
VI	Virtual Instrument
ViSARD	VHDL-integrated Softcore Architecture for Reconfigurable Devices
WDF	Wellen-Digital-Filter
WELMEC	Western European Legal Metrology Cooperation
ZEfIRA	Zertifizierbarer Entwicklungsprozess für Informationsverarbeitungssysteme mit speziellen Rechnerarchitekturen

Danksagung

Nach Jahren intensiver Arbeit ist die Dissertation nie das Werk einer einzelnen Person. Deswegen möchte ich mich an dieser Stelle bei denjenigen bedanke, die mich in dieser spannenden wissenschaftlichen Tätigkeit begleitet haben.

Ein besonderer Dank geht an meinen Doktorvater Prof. Wolfgang Fengler für die Möglichkeit, am Fachgebiet Rechnerarchitektur und Eingebettete Systeme der Technischen Universität Ilmenau zu arbeiten und promovieren zu können. Außerdem bedanke ich mich, dass er mir mit seinem Fachwissen immer zur Seite stand, viele seiner Feierabende für mich opferte und mir durch Gespräche zu einem wertvollen und freundschaftlichen Wegbegleiter wurde.

Für die Gelegenheit, an den Arbeiten im Bereich der dynamischen Wägetechnik im Fachgebiet Prozessmesstechnik beteiligt sein zu dürfen, möchte ich dem Prof. Thomas Fröhlich und seinen Mitarbeitern Hanna Baumgartl, Norbert Rogge und Gunter Krapf danken. Sie waren mir stets sehr gute Ansprechpartner und bereicherten meine Forschung durch ihre Anregungen und Ideen.

Des Weiteren bin ich dem Prof. Klaus-Dietrich Kramer für sein zweites Gutachten zu Dank verpflichtet. Ihre Einladung nach Wernigerode zur wissenschaftlichen Diskussion war wertvoll und sehr konstruktiv.

Ebenso geht mein Dank an meine wertvollen Kollegen Prof. Detlef Streitferdt, Dr. Bernd Däne, Dr. Oswald Kowalski, Dr. Todor Vangelov, Johannes Klöckner, Dr. Marcus Müller, Dr. Arvid Amthor, Andriy Osadchuk, Hans-Christian Schwannecke, Heiko Weiß, Dr. Patrick Mäder, Dr. Alexander Pacholik und Nils Würfel für die ausgezeichnete Zusammenarbeit, den vielen gemeinsamen Projekten und Diskussionen. Sie haben mir stets Mut zugesprochen und mich in meiner Arbeit bestärkt.

Den vielen Studenten, die ich während ihrer Studienzeit bei Projekt- und Abschlussarbeiten im Rahmen meiner Tätigkeit im Fachgebiet Rechnerarchitektur und Eingebettete Systeme betreute, gilt mein besonderer Dank. Sie leisteten einen großen Beitrag zu den erfolgreichen Untersuchungen und den Ergebnissen dieser Arbeit.

Eine herausragende Stellung in jeglicher Hinsicht nehmen meine Familie und mein lieber Mann ein, die mich in den vergangenen Jahren vielseitig unterstützt haben. Ich danke meinen Eltern, die die Grundsteine und sehr viel Hoffnung für meinen Weg gelegt haben. Lieber Robert, ohne deine liebevolle Fürsorge wäre diese Arbeit nicht zu dem Werk geworden.

1 Einleitung

Zunehmend spielt die Informationsverarbeitung eine wichtige Rolle in modernen Erzeugnissen. Diese Position wird auch durch permanente Forschung im Bereich der Messtechnik vorangetrieben. Von besonderem wissenschaftlichem Interesse sind dabei sehr komplexe Mess- und Automatisierungsalgorithmen, die effektiv realisiert und implementiert werden müssen. Es werden vor allem hohe Anforderungen an Qualität, messtechnische Eigenschaften, Verarbeitungsleistung, Echtzeitverhalten, Sicherheit und Kosten der Systeme gestellt.

In den letzten Jahren steigerte sich ständig die Komplexität der zu entwickelnden Informationsverarbeitungssysteme, so dass die verwendeten Entwicklungsmethoden verbessert und optimiert werden müssen. Dabei sind Eingebettete Systeme aus Software und Hardware heutzutage auf vielen Gebieten der Stand der Technik. Unter einem Eingebetteten System (engl. *embedded system*) versteht man ein Gerät, in dem ein Rechner integriert ist, wobei dieser weitgehend unsichtbar für den Anwender ist. Dabei ist dieses die Komponente eines technischen Systems, das über Sensoren und Aktoren mit einem Einbettenden System, z.B. den elektrisch-mechanischen Komponenten eines Messgeräts, interagiert (Abb. 1.1). Für die Entwicklung von solchen Systemen befasst sich die Forschung mit Prozessen, Methoden, Techniken und Werkzeugen zum Software-, Hardware- und Hardware-Software-Co-Entwurf (auch Hardware-Software-Codesign). [GM07]

Der Stand der Technik für eingebettete Hochleistungsplattformen für die Informationsverarbeitung wird zukünftig vor allem auch durch rekonfigurierbare Hardware charakterisiert werden, deren Einsatz modulare, flexible, leistungsstarke Systemkompositionen erlaubt [MB07]. Durch die Möglichkeit, mit hoher Parallelität in diesen Hardwarestrukturen zu arbeiten, sind Lösungen mit ihnen zumeist den sequentiell arbeitenden Prozessoren überlegen, selbst wenn diese als Parallelarchitekturen (Multicorearchitekturen oder ähnliches) betrieben werden.

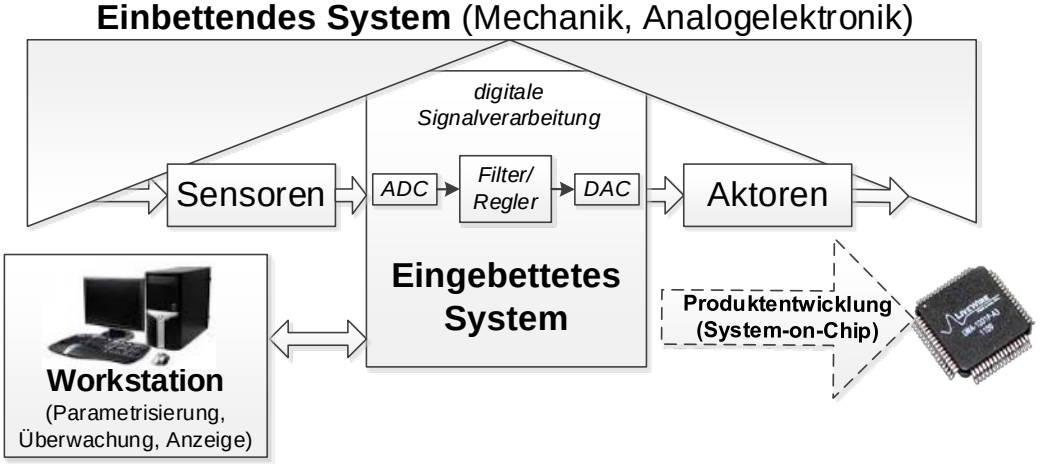


Abbildung 1.1: Beispielhafte Einordnung eines Eingebetteten Systems

Um eine systematische Entwicklung, und damit verbunden die Fehlervermeidung und Maßnahmen für die möglichst frühzeitige Fehlererkennung und –beseitigung beim Entwurf von komplexen Informationsverarbeitungssystemen zu fördern, werden Entwicklungsprozesse gebraucht. Dies ist auch für die messtechnischen mechatronischen Systeme mit ihrer hohen Dynamik bei Entwicklung, Komplexität und Technologievielfalt eine immer wieder neue Herausforderung. Bei der gemeinsamen gleichzeitigen Entwicklung von Eingebetteten und Einbettenden Systemen regeln die Entwicklungsprozesse die Technikorganisation von Arbeitsabläufen in der Hardware- und Software–Entwicklung und die in diesem Fall notwendigen Schritte, genauso wie bei der Integration von entwickelten Teilen in das Gesamtsystem. Es existieren Abhängigkeiten zwischen der Entwicklung von Eingebettetem und Einbettendem System.

1.1 Aufgabenstellung und Motivation

Eingebettete Systeme mit steigendem Funktionsumfang erobern immer neue Anwendungsgebiete in der industriellen Messtechnik. Einen besonderen Platz nimmt die Wägetechnik ein, die zu den ältesten Gebieten (mehrere 1000 Jahre) zählt.

Waagen haben eine große Bedeutung für die Massebestimmung in vielen Bereichen und unterliegen in den meisten Einsatzfällen der staatlichen Gesetzgebung. Im Vergleich mit anderen Messgebieten liegen die Anforderungen an die Einhaltung der geforderten Messunsicherheit in der Wägetechnik oft wesentlich höher. Eine entsprechende Rolle spielen Waagen bei der Automatisierung von Produktionsprozessen, besonders die selbsttätigen Waagen, die nach einer vorgegebenen Rezeptur verschiedene Komponenten zusammenführen können [Kr04].

Weiterhin sind die bisher eingesetzten Lösungen der Informationsverarbeitung nicht ausreichend, um Messunsicherheit und Dynamik deutlich zu verbessern. Die Qualität des wägetechnischen (mechatronischen) Teils und deren Steuerung und Signalverarbeitung beeinflussen das Endprodukt maßgeblich. Es ist eine sorgfältige Planung mit eindeutig definierten Schnittstellen unerlässlich. Auf dem Gebiet der Präzisionswaagen und Wägezellen für industriellen Einsatz sind viele Probleme erst ansatzweise erforscht. Die Wägezellen benötigen für ihre Funktion neue Ansätze zur komplexen und hochleistungsfähigen Informationsverarbeitung, die mit Eingebetteten Systemen realisiert werden können.

Die Lösungen für eingebettete Hardware-Software-Systeme müssen die folgenden wichtigen Funktionen realisieren: Unterstützung der Gewichtsermittlung, Signalverarbeitung einschließlich Regelung und/oder Steuerung. Weitere wichtige Kriterien für die Entwicklung des Systems sind Bedienelemente, Anzeigen, sowie Schnittstellen zur Kommunikation mit anderen Systemkomponenten. Basierend auf den oben genannten Anforderungen beschäftigt sich die vorliegende Arbeit mit der Erforschung eines qualifizierten Entwicklungsprozesses für derartige Informationsverarbeitungssysteme unter Berücksichtigung

der in der Wägetechnik geforderten Eichfähigkeit. In einer Reihe von Veröffentlichungen wird in diesem Zusammenhang von notwendiger Metrologischer Sicherheit gesprochen (z.B. in [BS10]), wobei der Begriff nicht klar definiert ist. Für die vorliegende Arbeit wurde dazu eine Definition erarbeitet, die im Kapitel 3.5 enthalten ist. Der eben genannte Prozess soll die gesamten Abläufe bei der Entwicklung der Informationsverarbeitung regeln, die Möglichkeit enthalten schnell und flexibel auf die messtechnischen Anforderungen zu reagieren und eine weitgehend optimale Entwurfslösung für das konkrete Messsystem zu erzeugen.

Es existieren mehrere wirtschaftliche, gesetzliche und technische Gründe für einen qualifizierten Entwicklungsprozess. Eine besonders wichtige Rolle spielen solche Prozesse in der Lebensmittel-, chemischen und pharmazeutischen Industrie, die außergewöhnlich hohe Anforderungen an Metrologische Sicherheit und Eichfähigkeit stellen. In diesem Zusammenhang wird auch ein systematischer Entwicklungsprozess notwendig, der zertifiziert sein sollte.

In den USA existiert eine Behörde, die „Food and Drug Administration“ (FDA), die für die Arzneimittelzulassung und die Lebensmittelüberwachung zuständig ist. Eine wichtige Aufgabe der FDA ist die Erarbeitung von Vorschriften und die Durchführung von Kontrollen der Herstellung von Medizinprodukten und Lebensmitteln. Es werden durch die FDA u. a. Forderungen an die verwendete Wägetechnik bei der Herstellung von Arznei- und Nahrungsmitteln gestellt, die von den Herstellern von Messgeräten für diese Prozesse u.a. auch beim Export in die USA realisiert werden müssen. Eine Forderung ist die Anwendung eines definierten und dokumentierten Entwicklungsprozesses. Ähnliches gibt es auch in anderen Ländern, wobei die Vorschriften und Kontrollen der FDA die systematischsten und konsequentesten Forderungen stellen [FDA14].

In Deutschland ist die Regelung, welche Messgeräte der Eichpflicht unterliegen, in Rechtsvorschriften geregelt: Gesetze (Eichgesetz, Einheitengesetz), Verordnungen/Vorschriften über die Eichpflicht von Messgeräten [ME09] und Eichpflichtausnahmeverordnung, Richtlinien (PTB - Physikalisch-Technische Bundesanstalt Prüfregeln [PTB14]).

Die zunehmende Rationalisierung und Automatisierung hat Einfluss auf die Qualitätserhöhung und die Kostenreduzierung der Erzeugnisse. Der qualifizierte Entwicklungsprozess hat nicht nur die Aufgabe, eine effektive Entwicklung zu ermöglichen, sondern auch auf gesetzliche Anforderungen reagieren zu können. Weiterhin sollen wirtschaftliche Ziele, wie die Senkung von Entwicklungskosten und die damit verbundene Steigerung des Gewinns verfolgt werden. Ein zusätzlicher wirtschaftlicher Aspekt der Entwicklung eines qualifizierten Prozesses ist die Betrachtung und die Integration von Aufgaben des Projektmanagements. Dieses soll sicherstellen, dass die entwickelten Messsysteme und die gestellten Projektziele die technischen, finanziellen, personellen und zeitlichen Randbedingungen erreichen. [Be05]

1.2 Ziele des Forschungsvorhabens

Das Hauptziel der Arbeit ist die Untersuchung und Entwicklung von Entwurfsmethoden, die für die Anforderungen der Messtechnik (im Weiteren am Beispiel der Wägetechnik) speziell weiterentwickelt, entworfen und angepasst sind. Diese werden in einen in der vorliegenden Dissertation erarbeiteten Entwicklungsprozess für die komplexe Informationsverarbeitung von Waagen mit eingebetteten Hardware-Software-Systemen integriert. Der Entwicklungsprozess soll die speziellen Anforderungen an Metrologische Sicherheit und Eichfähigkeit, gewährleistet durch Qualitätssicherung sowie durch andere Maßnahmen, schon zur Entwurfszeit auf Modellebene berücksichtigen und bis zur Implementierung verfolgen. Dabei sind Validier- und Verifizierbarkeit als Entwurfskriterien im gesamten Prozess zu integrieren. Dafür soll die Gewährleistung der Metrologischen Sicherheit mit ihren speziellen Qualitätsanforderungen für eingebettete Informationsverarbeitungssysteme der Wägetechnik definiert und für Hardware-Software-Systeme als Bearbeitungsgegenstand untersucht werden.

Sehr anspruchsvolle Lösungen im Bereich der Wägetechnik sind entweder dadurch gekennzeichnet, dass sie mit sehr hohen Auflösungen bei sehr kleinen Messunsicherheiten arbeiten oder in industriellen Prozessen eingesetzt werden. Im zweiten Fall sollen sehr schnelle Messungen in einem mechanisch gestörten Umfeld durchgeführt werden. In beiden Fällen bewegt man sich in Grenzbereichen der Wägetechnik. Um diese Grenzen zu noch kleineren Messunsicherheiten beziehungsweise zu einem schnelleren dynamischen Wägen unter starken und auch veränderlichen Störungen zu verschieben, ist eine Grundlagenforschung notwendig. Hierbei steht die Untersuchung von Prinzipien und Methoden im Vordergrund, die notwendig sind, um diese Ziele zu erreichen.

Zur Gewährleistung von korrektem Messen sollen aufwändige Algorithmen der komplexen digitalen Signalverarbeitung entwickelt und deren optimierte Implementierung untersucht werden. Ein Verschieben der messtechnischen Grenzen, wie oben genannt, zieht unmittelbar höhere Leistungsanforderungen sowie kürzere Reaktionszeiten der Informationsverarbeitung nach sich. Es werden Eingebettete Systeme notwendig sein, die eine entsprechende Rechenleistung, Reaktionszeit und Kommunikationsleistung realisieren. Die optimalen Lösungen bezüglich dieser Parameter für die nötigen eingebetteten Hochleistungsplattformen werden, vor allem durch rekonfigurierbare Hardware charakterisiert. Diese können die eben genannten Eigenschaften realisieren und sind damit im Hochleistungssektor in diesem Gebiet angesiedelt. Für die Beherrschung des Entwurfs solcher Systeme sind modellbasierte Methoden erforderlich, da sie zur Entwurfszeit die Optimierung der Plattformstruktur und der Funktionsverteilung erlauben, die von entscheidender Bedeutung für die Leistungsfähigkeit des Gesamtsystems sind.

Die Anforderungen sollen bereits frühzeitig im Entwicklungsprozess auf Modellniveau berücksichtigt werden. Dann sind optimale und fehlerärmere Lösungen möglich, wobei

sich gleichzeitig die Entwicklungszeiten verkürzen. Weiterhin soll die Optimierung der Leistungskennwerte im Sinne der Anforderungen unter Berücksichtigung der Restriktionen durchgeführt werden. Dadurch soll erreicht werden, dass das realisierte System alle notwendigen Funktionalitäten im Sinne der Spezifikation und der geforderten Systemparameter erfüllt und gleichzeitig eine möglichst wirtschaftlich günstige Lösung, sowohl für den Entwicklungsprozess als auch für das Zielerzeugnis entsteht. Deshalb sollen geeignete Methoden ausgewählt beziehungsweise entwickelt und in den Prozess integriert und validiert werden. Dabei soll dieser definiert und seine wichtigen Schritte erarbeitet werden. Es werden insbesondere betrachtet:

- Anforderungsanalyse und Testen von Anforderungen;
- Anforderungen der Eichfähigkeit und Metrologischen Sicherheit;
- Modellbasierter Entwurf einschließlich Verifikation und Validierung der Modelle und der eventuellen Nutzung von Modellen des Einbettenden Systems;
- Berücksichtigung der Eichfähigkeit und Metrologischen Sicherheit auf Modellebene;
- Implementierung in Hardware, rekonfigurierbare Hardware und Software (z.B. über Softcore-Prozessoren) als optimale Lösung für konkreten Anwendungsfall;
- Ableitung von Testfällen für das implementierte System aus den Modelltestfällen.

Bei der Eichfähigkeit von Software in eingebetteten Informationsverarbeitungssystemen gibt es bereits gesetzliche Regelungen und Empfehlungen [PTB06][WEL14]. Das gilt für die Metrologische Sicherheit jedoch noch nicht. Für rekonfigurierbare Hardware (FPGAs) sind zurzeit beide Probleme nicht gelöst und sind für diese Realisierungsform Bearbeitungsgegenstand der Dissertation.

Als letzte Etappe der Arbeit soll der Nachweis der Realisierbarkeit und der Effektivität des entworfenen Entwicklungsprozesses an einem Beispiel aus dem messtechnischen Bereich durchgeführt werden. Aus diesem Grund beschäftigt sich die Dissertation mit der Entwicklung eines Prototyps, der als Demonstrator für die Informationsverarbeitungsaufgaben in einem Messsystem eingesetzt wird.

1.3 Inhalt der Arbeit

Diese Arbeit ist wie folgt gegliedert. Das zweite Kapitel widmet sich dem Stand der Technik zu ausgewählten Aspekten. Dabei werden die allgemeinen Grundbegriffe und Definitionen aus dem messtechnischen Gebiet dargestellt. Die bestehenden Prinzipien der digitalen Informationsverarbeitung einschließlich spezifischer gesetzlicher und Sicherheitsanforderungen werden näher betrachtet und analysiert. Anschließend wird auf die

unterschiedlichen Arten von bekannten Vorgehensweisen zur Entwicklung von Eingebetteten Systemen eingegangen. Vorhandene Techniken des modellbasierten Entwurfs mit begleitenden Verifikations- und Validierungsaktivitäten werden detailliert dargestellt. Bezogen auf den plattformspezifischen Entwurf werden Informationen zu verschiedenen Implementierungsarten und eine besondere Betrachtung der FPGA-basierten Eingebetteten Systeme dargestellt. Der Abschnitt 2.5 fasst die Forschungsvorgaben der vorliegenden Dissertation aus den Untersuchungen zum Stand der Technik zusammen. Darauf aufbauend erfolgt im dritten Kapitel die Formulierung des zertifizierbaren Prozesses (ZEfIRA) für die Informationsverarbeitung in messtechnischen Systemen. Es wird eine spezielle Vorgehensweise für die Entwicklung von Eingebetteten Systemen präsentiert. Die Berücksichtigung der Metrologischen Sicherheit und der Eichfähigkeit sowie die Integration der Validierung und Verifikation im gesamten Entwurfsprozess werden erläutert.

Als Schwerpunkt der Arbeit werden im Kapitel 4 die Prozessschritte der effektiven Prototypenentwicklung nach ZEfIRA genauer beschrieben. Ausgehend von der werkzeugunterstützten Anforderungsanalyse bis zur Implementierung mit dazugehörigen Verifikations- und Validierungsaktivitäten wird das Eingebettete System (Informationsverarbeitungssystem) abhängig von den Forderungen des Einbettenden Systems (Messsystem) modellbasiert entwickelt. Die Methoden, Prinzipien und Techniken des durchgängigen Entwurfes, die im Kapitel 4 formuliert werden, unterstützen vorwiegend die Entwicklung von komplexen FPGA-basierten Eingebetteten Systemen. Die Anwendung des erarbeiteten Konzeptes der systematischen Entwicklung in einem Projekt wird im Kapitel 5 gezeigt. Anhand des prototypischen Entwurfs eines Informationsverarbeitungssystems in der elektromagnetischen Kraftkompensationswaage werden eine Leistungssteigerung und eine effektive Entwicklung, bezogen auf den personellen und Zeitaufwand, im Vergleich zum vorgestellten Vorläufersystem demonstriert. Im Abschnitt 5.3 wird die Einschätzung der Effektivität von ZEfIRA gegeben, wobei ein abgeschätzter zeitlicher Verlauf im Gegensatz zu einem Standardentwicklungsprozess (nach dem V-Vorgehen) dargestellt wird. Abschließend wird im Kapitel 6 der in der Dissertation ansatzweise bearbeitete Aspekt zur Erweiterung des Konzeptes ZEfIRA für den Produktlinienentwurf präsentiert. Eine Zusammenfassung der Forschungsergebnisse, eine Wertung derselben und ein Ausblick zu zukünftigen Arbeiten beschließen diese Arbeit.

2 Allgemeine Aspekte der Informationsverarbeitung in der Messtechnik anhand der Wägetechnik

Im modernen Leben existiert praktisch kein Gebiet, in dem das Messen keine Anwendung findet. Das einfache Messen verfolgt uns täglich im Haushalt, beim Einkaufen und Autofahren. Dabei werden unterschiedliche messtechnische Geräte eingesetzt, die in der gewünschten Genauigkeit den Messwert abschätzen können. Das Messen in Natur, Technik und Handel mit vorgegebenen Parametern war und ist für die Entwicklung dieser Anwendungsgebiete von maßgeblicher Bedeutung und verlangt wissenschaftliche Untersuchungen.

Das Gebiet der Messtechnik befasst sich mit Geräten und Methoden zur Erfassung von verschiedenen physikalischen Messgrößen (z.B. elektrische Größen, Länge, Temperatur, Masse, Kraft usw.). Die Entwicklung von Messmethoden und Messsystemen, sowie die Prinzipien bei der Erfassung von Messdaten mit verringerter Messunsicherheit sind die wichtigsten Teilgebiete der Messtechnik [WW10][Am10].

Im Allgemeinen bietet die Messtechnik zur Realisierung eines Messverfahrens zwei sehr unterschiedliche Prinzipien an, die gleichberechtigt nebeneinander stehen: Ausschlagprinzip und Kompensationsprinzip. Beim Ersten wird die Messgröße direkt oder indirekt in mehreren Stufen in die Ausgangsgröße umgeformt, dementsprechend liegt eine geradlinige Messkettenstruktur vor. Beim Kompensationsprinzip wird die Wirkung der Messgröße durch Ausgleichen mit einer gleichartigen stellbaren Kompensationsgröße ermittelt. Dabei ist die Messkette als kreisartige Struktur ausgelegt [WW10][PP94].

Das folgende Kapitel enthält die allgemeinen Aspekte der Informationsverarbeitung in der Messtechnik. Im Abschnitt 2.1 werden die wichtigsten Grundbegriffe und Definitionen des messtechnischen Gebietes dargestellt, die im Laufe der Jahre als eigene Fachsprache für exaktes Arbeiten und Beschreiben in Abhängigkeit von Naturwissenschaft und Technik entwickelt wurden. Die begrifflichen Festlegungen wurden unter anderem in [JCGM10][DIN1319] beschrieben.

In den Kapiteln zu den eigenen Forschungsergebnissen wird sich die Arbeit zumeist auf die Wägetechnik konzentrieren. Das liegt vor allem daran, dass in diesem Gebiet sehr viele Aspekte der Messtechnik in Kombination auftreten, wie z.B. dynamisches Verhalten und Eichfähigkeit. Es führt dazu, dass relativ komplexe Gesamtlösungen notwendig sind und die Entwicklung der Geräte ein systematisches Vorgehen verlangt. Darauf basierend werden in den folgenden Abschnitten zum Stand der Technik zumeist die Informationen zur Masse- und Gewichtsbestimmung detaillierter dargestellt.

2.1 Grundbegriffe und Definitionen in der Messtechnik

Der Vorgang des Messens hat zwei Aspekte. Einerseits wird die physikalische Größe erfasst und dargestellt, andererseits normiert oder einer Maßzahl zugeordnet. Dabei muss die zu messende Größe qualitativ eindeutig definiert und quantitativ bestimmbar sein. Das Messnormal muss dazu aber durch eine Konvention festgelegt werden. Eine Größe wird als Vielfaches oder Teil einer zu dieser Größe festgelegten Einheit angegeben. Größen, die durch eine Messung bestimmt werden, sind Messgrößen. Die voneinander unabhängig definierten Normale werden Grundnormale (oder Basiseinheiten) genannt. Durch die „Conference Generale des Poids et Mesures“ sind die sieben Grundnormale des internationalen Einheitssystems (SI) definiert: das Meter [***m***], das Kilogramm [***kg***], die Sekunde [***s***], das Ampere [***A***], das Kelvin [***K***], die Candela [***Cd***] und das Mol [***mol***]. [PP94]

Es existieren noch viele weitere Einheiten, die über physikalische Gesetze auf die Basiseinheiten zurückgeführt werden. Z.B. wird bei der Bestimmung der Masse eines Körpers zwischen Masse- und Kraftbestimmung unterschieden [Ko89].

Der folgende Absatz wurde wörtlich aus [Fin64] entnommen, da er diesen Sachverhalt sehr prägnant beschreibt:

> *„Die Masse eines Körpers ist die Gesamtheit seiner materiellen Teilchen. Diese Masse (Einheit = **kg**) ist konstant und unabhängig von dem Ort, an dem sich der Körper gerade befindet. Da sich der Körper mit seiner konstanten Masse gewöhnlich in einem Schwerefeld befindet, wird er von diesem angezogen und übt dabei eine Kraft (Einheit = **N** oder kp) auf seine Unterlage aus. Je nach dem Ort, wo sich der Körper gerade befindet, kann diese Kraft (oder auch Last) verschieden groß sein; denn die Schwerebeschleunigung ist nicht an allen Orten der Erdoberfläche gleich. Da wir aber einen Wert des Körpers ermitteln wollen, der überall gleich ist, müssen wir seine Masse bestimmen.“*

Die Einheit der Kraft wird über das zweite Newtonsche Axiom (***F=m*a***) definiert, als Maßeinheit ergibt sich dafür $\boldsymbol{kg\ m\ s^{-2}}$. Diese Einheit wird häufig gebraucht und erhält einen eigenen Namen: 1 Newton mit der Abkürzung ***N*** [WW10][Bo88].

Das Messen als Vergleich mit einem Normal wird nur in seltenen Fällen durchgeführt. Meistens werden Messgeräte verwendet, die die gewünschten Messwerte nach einem physikalischen Effekt anzeigen. Dabei werden unterschiedliche Messprinzipien gebraucht, um eine Messgröße in die Anzeige eines Messgerätes zu wandeln. Der quantitative Zusammenhang zwischen der Anzeige oder dem Ausgangssignal einer Messeinrichtung und dem wahren Wert der Messgröße über den gesamten Messbereich der Messeinrichtung wird durch Kalibrieren bestimmt [DIN1319]. Der eigentliche Messeffekt wird mit dem Hilfsmittel (heute zumeist Elektronik) auf einer Anzeige (analog - Skala mit Zeiger oder digital – mit Ziffern) als Messwert dargestellt. Dabei wird die Zusammen-

schaltung von Komponenten für die Wandlung einer nichtelektrischen Größe in ein elektrisches Signal bis zur Anzeige als Messkette bezeichnet [WW10]. An dieser Stelle muss die Definition für die Auflösung eines anzeigenden Messgerätes gegeben werden. Unter der Auflösung wird allgemein die erforderliche Änderung der Messgröße verstanden, um eine festgelegte Änderung der Anzeige zu bewirken [PP94][DIN1319]. In der digitalen Signalverarbeitung wird dieser Begriff im Zusammenhang mit der Quantisierung verwendet.

In der Wägetechnik gilt nach [Be96]:

> *„Die Waage ist ein Meßgerät für Massen. Die modernen elektronischen Waagen bestimmen die Masse über die Wirkung der Schwerkraft der Erde auf diese Masse, das heißt, sie messen die Gewichtskraft **G**.“… „Eine Waage kann auch eingesetzt werden, um andere, mit der Masse zusammenhängende Größen zu messen, z. B. Stückzählung von gewichtsgleichen Teilen oder Dichtebestimmung, oder um Kräfte zu messen, z. B. Oberflächenspannungen oder Federkräfte.“*

Das Ziel jeder Messung ist es immer eine genaue Bestimmung der Messgröße mit einer dem Problem angepassten Genauigkeit durchzuführen. Ohne Genauigkeitsabschätzung ist die eigentliche Angabe des Messwertes wenig aussagefähig. Grundsätzlich ist aber jede Messung mit Fehlern behaftet. Die Messgröße selbst aber auch die verwendeten Messverfahren und Messgeräte sind ungenau, sodass eine tatsächlich existierte Größe von ihrem gemessenen Wert abweicht. Diese Differenz wird als Messabweichung bezeichnet. Die Abbildung 2.1 stellt die unterschiedlichen Ursachen der Messabweichungen dar (angelehnt an [PP94][WW10]).

Abbildung 2.1: Entstehungsmechanismen für Messabweichungen

Da der richtige (wahre) Wert in der Praxis nicht ermittelt werden kann, führt die Genauigkeitsabschätzung zu einem Wahrscheinlichkeitsintervall. Dieses Intervall um ein Messergebnis, das den geschätzten Bereich angibt, innerhalb dessen der wahre Wert mit einer bestimmten Wahrscheinlichkeit liegt, wird als Messunsicherheit [DIN1319] bezeichnet.

Eine besondere Betrachtung ist notwendig, wenn mehrere Glieder der Messkette zum Messergebnis beitragen. Dabei liefern alle Messunsicherheiten dieser Glieder einen Beitrag zur Messunsicherheit des gesamten Messergebnisses. Dieses wird als Fehlerfortpflanzung bezeichnet. Der Einfluss der verschiedenen Glieder auf das Messergebnis kann von sehr unterschiedlichem Gewicht sein. Auf Grund der verschiedenen Art von systematischen (deterministischen) und zufälligen (stochastischen) Fehlern sind auch deren Fortpflanzungsgesetzte unterschiedlich. Die maximale Messunsicherheit ergibt sich durch Addition der abgeschätzten Größtwerte der unbekannten systematischen und der zufälligen Abweichung (auch Fehler in [PP94] genannt) der einzelnen Messglieder. Zur Schätzung der daraus resultierenden Messunsicherheiten verwendet man die zwei folgenden Definitionen [WW10]:

1. ***Absoluter Fehler*** Δx_i einer Einzelmessung ist gleich der Abweichung vom Mittelwert $\overline{x}$ aller n Messungen x_i mit $i = \{1,\ldots, n\}$
2. ***Relativer Fehler*** ist das Verhältnis vom absoluten Fehler zum Messwert: $\frac{\Delta x_i}{x_i}$

Systematische Fehler treten bei wiederholten Messungen unter gleichen Bedingungen immer mit dem gleichen Betrag und Vorzeichen auf. Diese lassen sich durch eine sorgfältige Untersuchung möglicher Fehlerquellen beeinflussen, z.B. reduzieren oder korrigieren.

Hingegen werden zufällige Fehler durch nicht vorhersagbare Störungen (zufällige Einflüsse während der Messung) verursacht. In der Messwertanalyse werden diese nach den Methoden der Statistik erfasst und behandelt. Es werden wiederholte Messungen durchgeführt, und die Menge von Einzelmessungen wird Stichprobe genannt. Aus den Einzelmessungen erhält man einen Schätzwert nach der mathematischen Grundlagen der Normalverteilung von Zufallsgrößen von Gauss [JCGM10][BST10][DIN1319].

Daraus folgend kann die beste Abschätzung des Messwertes x durch das arithmetische Mittel realisiert werden:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \textbf{(2.1)}$$

Dann ist der durchschnittliche (mittlere) Fehler bei der Einzelmessung:

$$\boldsymbol{\sigma^2 = (\Delta x)^2 = \frac{1}{n-1}\sum_{i=1}^{n}(\Delta x_i)^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2} \qquad \textbf{(2.2)}$$

Dabei ergibt sich als mittlerer Fehler (auch als Standardabweichung bezeichnet):

$$\boldsymbol{\sigma_{\overline{x}} = \Delta\overline{x} = \sqrt{\frac{1}{n(n-1)}\sum_{i=1}^{n}(x_i - \overline{x})^2} = \frac{\Delta x}{\sqrt{n}} = \frac{\sigma}{\sqrt{n}}} \qquad \textbf{(2.3)}$$

Die Messung von dynamischen Messgrößen, die sich zeitabhängig ändern, ist sowohl vom Messgerät als auch vom zeitlichen Verlauf des Messsignals abhängig. In diesem Fall spricht man über dynamische Messabweichungen der Messgröße. Diese Abweichungen sind zeitlich veränderlich. Aus diesem Grund benötigt man eine besondere Analyse des zeitlichen Verhaltens des Messsystems beziehungsweise des Messgerätes [Ja10]. Mathematisch kann man das zeitliche Verhalten mit Differenzialgleichungen beschreiben. Es ist theoretisch möglich, ein beliebiges Messsignal aus dem beobachteten Ausgangssignal zu rekonstruieren. In der Praxis werden zumeist Testfunktionen benutzt. Im Fall einer Sprungfunktion wird beispielsweise die Einstellzeit oder Einschwingzeit gemessen. Das ist die Zeitdauer, die gebraucht wird, bis die angezeigte Messgröße endgültig innerhalb eines gegebenen Toleranzbereiches verbleibt [PP94]. Eine andere Funktion ist ein sinusförmiges Eingangssignal variabler Frequenz. Dabei wird die Reaktion des Messsystems im Frequenzbereich untersucht [WW10]. Hier ist die sogenannte Grenzfrequenz des Messsystems als diejenige Frequenz einer harmonischen Anregung definiert, für welche das Amplitudenverhältnis zum Ausgangssignal festgelegte Toleranzwerte, in der Regel entspricht das ±3dB in der Bode-Darstellung [PP94], erreicht.

Die technischen Parameter und vor allem die messtechnischen Eigenschaften des Messsystems stehen immer im Vordergrund bei der Entwicklung. Für Messgeräte, die den zahlreichen internationalen und nationalen Vorschriften und Gesetzen entsprechen sollen, werden in diesen die messtechnischen Mindestanforderungen gestellt. Der folgende Abschnitt 2.2 stellt eine Zusammenfassung der Gesetze, Vorschriften und Richtlinien dar. Diese müssen bei der Entwicklung von derartigen Messgeräten, detailliert am Beispiel der Wägetechnik, berücksichtigt werden, wobei hier vorwiegend die für die danach folgenden Darstellungen wichtigen Aspekte betrachtet werden.

2.2 Eichfähigkeitsanforderungen: Gesetze, Vorschriften und Richtlinien

Wenn es um Messgeräte geht, spricht man von Messbeständigkeit. Die Funktionstüchtigkeit muss während deren gesamten Lebensdauer gewährleistet sein [DIN1319]. In der vorliegenden Arbeit werden geeichte Messgeräte (am Beispiel der Wägetechnik) betrachtet, deren Anforderungen an die Richtigkeit und Messbeständigkeit durch das Eichgesetz festgelegt sind. Nach [DIN1319][WW10] wird das Eichen als Kalibrieren durch amtliche Institutionen definiert. Dabei wird eine Prüfung des Messgerätes durchgeführt, ob seine Beschaffenheit und seine messtechnischen Eigenschaften den Forderungen der Eichvorschriften genügen. Insbesondere wird nachgewiesen, dass die Beträge der Messabweichungen die Fehlergrenze nicht überschreiten. Anschließend wird die Stempelung des Messgerätes durchgeführt. Durch diese wird beurkundet, dass das Messgerät zur Prüfungszeit den Eichfähigkeitsanforderungen genügt und bei einer Handhabung innerhalb der Nacheichfrist „richtig“ bleibt.

Waagen werden seit vielen Jahren im eichpflichtigen Verkehr eingesetzt. Dabei werden sie in Genauigkeitsklassen eingeteilt, welche beispielsweise Mindestlast und Fehlergrenzen bestimmen [Be96]. Die Zulassung eines Wägesystems erfolgt nach der Eichpflicht in zwei Schritten. Zuerst wird ein Produktionsmuster vom Hersteller zur Bauprüfung an die Physikalisch-Technische Bundesanstalt [PTB14] eingereicht. Im zweiten Schritt wird jede individuelle Waage auf die Einhaltung der gesetzlichen Fehlergrenzen geprüft. In diesem Zusammenhang müssen die folgenden Anforderungen erfüllt werden [Ko89]:

- Richtigkeit des Messergebnisses (innerhalb der Fehlergrenze);
- Beständigkeit des Messwertes während der Eichgültigkeitsdauer;
- Sicherheit der Arbeitsweise unter dem Einfluss von Störgrößen;
- Ständige Einsatzbereitschaft der Waage;
- Gleichbehandlung aller Bauarten hinsichtlich der gestellten technischen Anforderungen ohne Beachtung des physikalischen Wirkungsprinzips;
- Harmonisierung in der europäischen Gemeinschaft (s. unten OIML [O14])

Nach erfolgreicher Prüfung kann die geeichte Waage für die Gültigkeitsdauer der Eichung im eichpflichtigen Verkehr verwendet werden. Nach Ablauf dieser Frist erfolgt die Nacheichung [Be96].

Die internationale Harmonisierung wird auf europäischer Ebene durch EU-Richtlinien vorangetrieben. Für die Messgeräte ist die allgemeine Richtlinie 2004/22/EG zuständig [ME09]. Diese wird als Messgerätenrichtlinie oder als Measurung Instruments Directive (MID) bezeichnet. Zusätzlich wird in der Wägetechnik die Richtlinie 2009/23/EG von der PTB gefordert [Am09].

In [Tr15] steht der Zweck der Harmonisierung:

> *„Die weltweite Vereinheitlichung der Vorschriften im Bereich des Eichwesens hat eine Doppelfunktion: auf einer Seite die Schaffung international anwendbarer, wiederholbarer wissenschaftlicher Maßeinheiten und zweitens die Gewährleistung der Einhaltung dieser Maße und Gewichte sowie der diesbezüglichen gesetzlichen Vorschriften in der Praxis des internationalen Warenverkehrs“.*

Es existieren die internationalen Empfehlungen der „International Organisation of Legal Metrology“ (OIML). Zu metrologischen Aspekten nichtselbsttätiger Waagen gibt es die europäische Norm EN 45501 [DN92]. Diese ist nahezu gleich mit der internationalen Empfehlung R76 der OIML.

Eine weitere europäische Vereinigung der nationalen Behörden, die im gesetzlichen Messwesen tätig sind, ist WELMEC (Western European Legal Metrology Cooperation). Die WELMEC-Dokumente und Leitfäden stellen den Stand der Technik dar, haben aber nur einen empfehlenden Charakter [WEL14]. Im Abschnitt 2.3.2 werden die Empfehlungen von WELMEC bezüglich der Informationsverarbeitung in Messsystemen detaillierter dargestellt.

Neben der nationalen Eichbehörde gibt es in anderen Länder auch Organisationen, die zusätzliche Forderungen für Messgeräte aufstellen. In der Wägetechnik müssen die Waagen, die in Medizin- oder Lebensmittelindustriebereichen eingesetzt werden, auch andere besondere Bestimmungen erfüllen. Dazu zählt man die Richtlinien und die Zertifizierungsvorschriften der FDA [FDA14]. Außerdem werden von dem Hersteller des Messgerätes die detaillierte Beschreibung aller Entwicklungsphasen und die Spezifikation des Produktes abgefragt. Diese werden normalerweise in Form eines Entwicklungsprozesses dokumentiert, und durch die FDA wird das Messgerät nach den beschriebenen Etappen überprüft. Die Hauptphasen des zertifizierten Entwicklungsprozesses sind dabei: Anforderungen und Spezifikation, Entwurf und Entwicklungsplanung, Verifikation und Validierung, Fertigung, Verpackung und Kennzeichnung, Kommerzialisierung. [MJZ13]

Diese und noch weitere spezielle Software- und Hardware-Standards müssen bei der Entwicklung von Messsystemen (hier Waagen) berücksichtigt werden. Es wird dabei nicht nur die Einhaltung von bestehenden Eichvorschriften und Sicherheitsvorschriften, sondern auch die Gewährleistung eines kontinuierlich ablaufenden Produktionsprozesses gefordert. Deswegen müssen diese bereits bei der Entwicklung von Prototypen integriert werden. In dieser Arbeit wird ein Weg vorgestellt, welcher die systematische Abarbeitung der einzelnen Forderungspunkte beschreibt.

2.3 Digitale Informationsverarbeitung in der Messtechnik

Die Informationsverarbeitung in der Messtechnik besteht nicht nur aus der Signalverarbeitung in einem Messsystem, sondern auch aus der Regelungs- und Steuerungstechnik,

Datenübertragung, Speicherung und Messdatenaufbereitung. Diese Aufgaben können mit analoger Elektronik oder unter Nutzung von digitaler Technik realisiert werden [Ja10].

Die analoge Informationsverarbeitung benutzt zumeist den zeitlichen Verlauf eines Signals (abgebildet auf eine Spannung) als Darstellung für Mess- und Zwischengrößen. Dabei existieren die folgenden Anforderungen an die analoge Elektronik [Hü96][Ja10]:

- Realisierung von Filterfunktionen, einfachen Regelalgorithmen, elementaren arithmetischen Operationen und Übertragung elektrischer Signale;
- Realisierung der Funktionen in den geforderten Genauigkeiten und in dem notwendigen Zeitverhalten.

Dabei treten die für analoge Elektronik bekannten Probleme auf [PP94][Hü96][Ja10]:

- geringe Funktionsvielfalt;
- nicht veränderbare Funktionsfestlegung durch Verdrahtung;
- Nichtlinearitäten des Ausgangs-/Eingangsverhaltens über den Darstellungsbereich des Signals;
- Frequenzabhängigkeit des Ausgangs-/Eingangsverhaltens;
- begrenzte Auflösung;
- Störempfindlichkeit (z.B. durch elektromagnetische Felder oder durch Beeinflussung der elektrischen Signale untereinander);
- Temperaturveränderung;
- Signalrauschen;
- Fertigungstoleranzen der Bauelemente;
- Begrenzte oder fehlende Möglichkeiten zur Abspeicherung der Signalwerte;
- Spezielle konstruktive Probleme (z.B. Platzbedarf, Stromaufnahme und Wärmeentwicklung).

Im Gegensatz zur analogen Signaltechnik werden in der digitalen Signalverarbeitung kontinuierlich verlaufende Signale in diskrete Signalwerte kodiert. In diesem Fall muss jede kodierte Größe durch mehrere geordnete Binärsignale beschrieben werden [WW10]. Die digitale Darstellung und Verarbeitung von Informationen benötigt deshalb die Analog-Digital-Wandlung der Eingangs- und gegebenenfalls die Digital-Analog-Wandlung der Ausgangssignale. Eventuelle Nachteile, die durch eine kodierte Darstellung entstehen, werden jedoch in dem messtechnischen Bereich durch die Vorteile der digitalen Informationsverarbeitung mehr als kompensiert [Sa12].

Verfahren der digitalen Informationsverarbeitung existieren bereits seit den letzten Jahrzehnten. Grundlagen von numerischen Techniken wurden schon im 19. Jahrhundert verwendet. Hier kann man die bekannte mathematische Theorie *(Fourier-Analyse oder klassische harmonische Analyse)* von Fourier nennen, die es ermöglicht, eine Zeitfunktion in eine Fourier-Reihe zu zerlegen und somit deren Frequenzspektrum diskret darzustellen [Br96]. Eine weitere wichtige Erfindung ist die Theorie der Signalabtastung von Nyquist und Shannon (in der russischen Literatur: Abtasttheorem von Kotelnikow) [BR13]. In der Mitte und am Ende des 20. Jahrhunderts verursachte die rasante Entwicklung der Rechentechnik eine stark zunehmende Bedeutung der digitalen Informationsverarbeitung in verschiedensten technischen Bereichen. Eines dieser Anwendungsgebiete ist seit längerer Zeit die Messtechnik [PP94], wobei mehrere wesentliche Vorteile auftreten [GM07].

Mit der digitalen Informationsverarbeitung wird es möglich komplexe Algorithmen mit einem hohen Maß an Flexibilität in kürzerer Zeit berechnen zu können. Dabei wird eine Software- und Hardware-gesteuerte Verarbeitung mit der Rechentechnik realisiert. Außerdem sind verschiedene Strategien der Signalverarbeitung möglich, z.B. in der Wägetechnik - nichtlineare oder adaptive Regelung im Gegensatz zur klassischen Analogregelung [WGA13]. Die Funktionen sind durch Programmierung leicht änderbar. Nichtlinearitäten des Ausgangs-/Eingangsverhaltens verschieben sich zu deutlich höheren Frequenzen [Ja10]. Als Weiteres ist die Darstellung von Daten mit deutlich höherer und anpassbarer Genauigkeit möglich.

Als einen Vorteil kann man auch hohe Sicherheit nennen. In [Sa12] steht:

> *„Während sich die analoge Information durch jede - auch noch so kleine - Störung verändert, wird die digital kodierte Information erst dann verfälscht, wenn ein Störimpuls größer ist als der Störabstand des jeweiligen digitalen Pegels. Der Störabstand ergibt sich aus der Differenz von Sende- und Empfangspegel.… Durch die Wahl der binären Informationsdarstellung ist es möglich, den Störabstand in weiten Grenzen auf die Gegebenheit des Umfeldes abzustimmen“*.

Damit haben die Fertigungstoleranzen und die Temperaturabhängigkeiten der Bauelemente, das Rauschen und die gegenseitige Beeinflussung der Signale praktisch keinen Einfluss mehr.

Ein weiterer Vorteil ist die einfache Datenspeicherung. Außerdem sind kurz- und langzeitige Speicherungen möglich. Hiermit ist die Anforderung der Messtechnik, große Datenmengen sicher und preiswert zu speichern, effektiv realisierbar [Ja10].

Als besonders wichtig werden die vielfältigen Übertragungsmöglichkeiten in der digitalen Informationsverarbeitung genannt. Die digitalen Werte lassen sich einfach auf vielfältige Weise kodieren. Aus diesem Grund sind große Distanzen und geringer Aufwand bei der Übertragung möglich [PP94]. Bei den konstruktiven Parametern erweist sich

ebenfalls die digitale Realisierung zumeist als günstiger. Das gilt nicht unbedingt für deren Kosten [Be91].

Die bereits benannten Funktionen, Analog-Digital-Wandlung und Digital-Analog-Wandlung, benötigen eine hybride Realisierung und weisen damit Vor- und Nachteile beider Realisierungstechnologien (analog und digital) auf [Ja10][PP94][BST10].

In den folgenden Abschnitten werden bestehende Prinzipien der digitalen Informationsverarbeitung in der Messtechnik näher erläutert. Der Hauptgesichtspunkt der Arbeit liegt in der dynamischen Wägetechnik. Deswegen werden deren bekannte Lösungen im Bereich der Signalverarbeitung detaillierter dargestellt und im Abschnitt 2.3.2 die Besonderheiten unter Berücksichtigung der Eichfähigkeit und anderer Forderungen diskutiert.

2.3.1 Bestehende Prinzipien der digitalen Informationsverarbeitung in der Mess- und Wägetechnik

Die Aufgabe der Messtechnik besteht darin, die eindimensionalen und mehrdimensionalen Messgrößen technischer Prozesse aufzunehmen. Weiterhin werden die erhaltenden umgeformten Messsignale als Messwerte so verarbeitet, dass die Zielgröße (Messergebnis) gewonnen wird [Hü96]. In diesem Zusammenhang werden die speziellen Aufgaben eines mechatronischen Systems in fundamentaler Weise durch die Informationsverarbeitung der Eingangssignale (Sensor und Bediensignale) und durch die Ausgabe der Information (in Form von Messergebnissen oder als Stellsignale) bestimmt [Ja10].

Die Abbildung 2.2 stellt die Messglieder einer Signalverarbeitungskette in einem Messsystem dar.

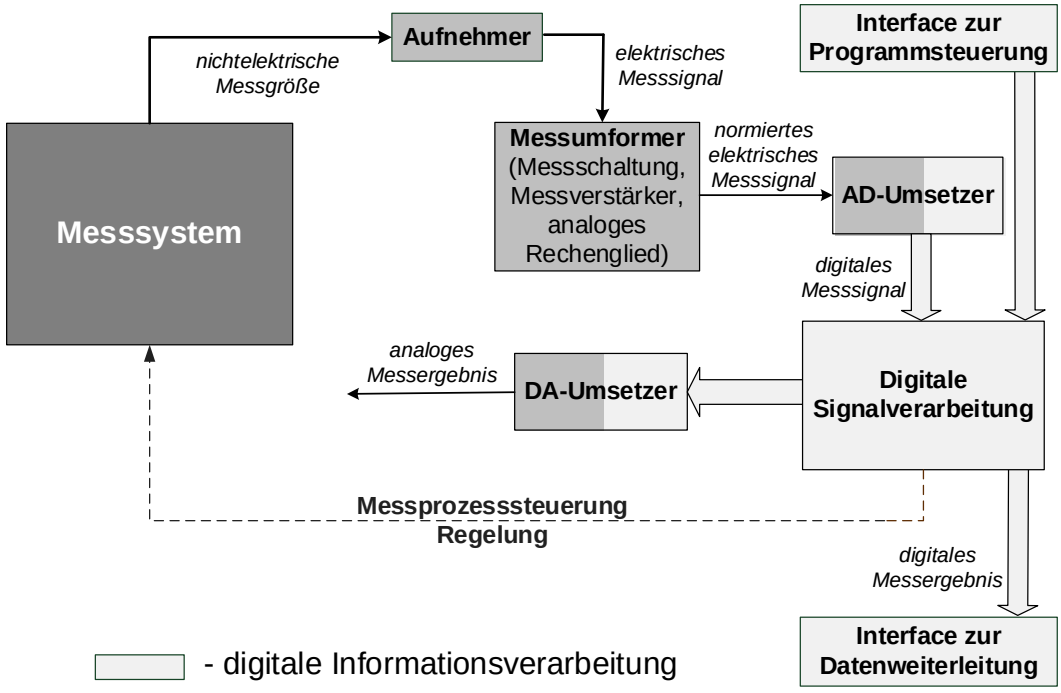


Abbildung 2.2: Signalverarbeitungskette mit digitaler Informationsverarbeitung

Zunächst formen Sensoren (Aufnehmer) die im Allgemeinen nichtelektrische Messgröße in ein elektrisches Messsignal um. Weiterhin wird das Signal dank geeigneter Messschaltungen, Messverstärker und anderer Rechenglieder umgeformt und ein normiertes analoges Messsignal gewonnen. In dem digitalen Informationsverarbeitungssystem wird die Analog-Digital-Umwandlung benötigt. Dabei wird das analoge Signal bezüglich Zeit und Wert zuerst abgetastet und quantisiert, und anschließend in ein digitales Messsignal kodiert [PP94]. Dabei muss es so bemessen werden, dass das resultierende Raster der diskreten Werte die Messgenauigkeit nicht beeinträchtigt [WW10].

Für jede Analog-Digital-Umwandlung wird eine gewisse Zeit gebraucht (abhängig von den Charakteristiken des Umsetzers). Darüber hinaus sorgt der begrenzte Speicherplatz für eine Begrenzung der zeitkontinuierlich aufgezeichneten Digitalwerte [HHR12]. Deswegen wird man dazu gezwungen, das zeitkontinuierliche Signal über einen bestimmten Zeitraum in regelmäßigen Abständen abzutasten. Es muss berücksichtigt werden, dass der Informationsgehalt des Messsignals wegen der Alias-Effekte verfälscht wird. Um diese zu vermeiden, muss die Abtastfrequenz mindestens doppelt so groß wie die höchste auftretende Signalfrequenz gewählt werden: $f_{Signal} < f_{Abtast} / 2$. Dieses Theorem wird in unterschiedlichen Literaturquellen verschieden genannt [PP94][Hü96][BR13], da aber Shannon 1949 eine informationstechnische Begründung dafür formulierte [WW10], wird es hier Nyquist-Shannon-Theorem genannt. In vielen Messsystemen treten häufig höhere Schwingungsfrequenzen auf. Damit diese Frequenzen unterdrückt werden oder die Abtastfrequenz in Bezug zur Grenzfrequenz möglich gering bleibt, wird ein analoger Tiefpassfilter vor dem Abtast-Halteglied (als Antialiasing-Filter bekannt [Hü96][WW10]) geschaltet.

Eine weitere Schnittstelle zwischen dem Informationsverarbeitungssystem und der analogen Umgebung im Messsystem stellt die Digital-Analog-Umwandlung dar. Abhängig von der zu ermittelnden beziehungsweise zu verarbeitenden Information wird diese digital und/oder analog ausgegeben. [PP94][HHR12]

Wenn Stellsignale für die Aktoren des Messsystems erzeugt werden, wird der Wirkungskreis geschlossen (Messprozesssteuerung, Regelung). Der Wirkungskreis hat die Aufgabe, unerwünschte Störeinflüsse im Gesamtsystem zu unterdrücken, gezieltes Bewegungsverhalten im Messsystem zu realisieren und robuste Stabilität bei Parameterunbestimmtheiten- und –variationen zu ermöglichen [Ja10][RZ04].

Die oben beschriebenen Messglieder der Signalverarbeitungskette sind nicht ideal, wodurch die Messsignale häufig verfälscht werden [Hü96][RZ04]. Für diese Fälle wird eine korrigierende Signalverarbeitung durchgeführt, damit das ermittelte Messergebnis mit einer kleineren Messunsicherheit bestimmt wird. Hier können die Korrekturen z.B. als Kalibrierung, Linearisierung oder digitale Filterung durch einen Messrechner durch-

geführt werden. Die Digitalfilter können in unterschiedlichen Topologien aufgebaut werden (FIR-Filter „finite impulse response filter“, IIR-Filter „infinite impulse response filter“, WDF „Wellen-Digital-Filter“). In der Messtechnik kommt häufig der digitale Mittelwertfilter zum Einsatz [Ja10][Do08], wobei von einem Signal ein „gleitender Mittelwert“ (moving average) gebildet wird. Laut [Be91] kann der Mittelwertfilter als FIR- und IIR-Filter realisiert werden. Grundsätzlich kann ein Digitalfilter hardwaremäßig beziehungsweise softwaremäßig umgesetzt werden. Die Implementierung durch Hardware ist die schnellste, jedoch sind die Kosten und die Realisierbarkeit von der Komplexität der Algorithmen abhängig. Die softwaremäßige Realisierung ist meist kostengünstiger, dabei wird der Digitalfilteralgorithmus durch einen Digitalrechner (z.B. Mikroprozessor, DSP) abgearbeitet [PP94][Ro91].

In der Praxis haben sich die Methoden von adaptiven Algorithmen bewährt [MB07]. Hierbei werden die Regler- und Filteralgorithmen an die störenden Einflüsse, sowie an die Messzeit im Signalverarbeitungssystem angepasst [Do08][Be91]. Es existiert eine Vielzahl von Algorithmen, die für die Adaptierung der Parameter verwendet werden können. Die Geschwindigkeit der Adaption ist von den eingesetzten Algorithmen und von den Realisierungsvarianten abhängig. Ein häufig verwendetes Beispiel eines adaptiven Filteralgorithmus ist der Kalman-Filter, der in mechatronischen Systemen als Störbeobachter beziehungsweise Zustandsbeobachter eingesetzt wird. Dabei können die nicht messbaren Störungen im System geschätzt werden [Am10][RZ04].

In dieser Arbeit liegt der Schwerpunkt auf der digitalen Informationsverarbeitung in der dynamischen Wägetechnik. Der Anspruch der dynamischen Wägesysteme ist, dass sie in automatisierte Produktionsprozesse eingebunden werden und diese dabei wenig beeinträchtigen. Beim dynamischen Wägen befindet sich das Wägegut in Bewegung, wodurch die Aufgabe der Gewichtsermittlung erschwert wird. Es muss ein Kompromiss zwischen Durchsatzleistung und Genauigkeit gefunden werden. Die Forderung der Messunsicherheit nimmt einen größeren Stellenwert ein, demzufolge muss die Geschwindigkeit des Wägeablaufes optimal gewährleistet werden. Für einen höheren Durchsatz von Wägegütern wird gefordert, schnellere, genauere und störsicherere Waagen einzusetzen [Kr04]. In der Praxis erweisen sich die Waagen nach dem Prinzip der elektromagnetischen Kraftkompensation (EMK) als sehr gut geeignet [Ba15]. Nach dem Stand der Technik werden diese Waagen heutzutage zumeist für hochpräzise Aufgaben in zeitkritischen Prozessen eingesetzt. [BS91][Jä94]. Es sind aber auch Wägezellen mit Dehnungsmessstreifen (DMS-WZ) für die dynamische Wägung unter industriellen Bedingungen verbreitet [Jä94]. Die DMS-WZ sind üblicherweise auf Anwendungen geringer bis mittlerer Präzision beschränkt. Mehr Information findet man unter anderem in [Mä09]. Diese Dissertation von Mäuselein untersuchte und entwickelte einen neuartigen Sensor für Einsatzbereiche mit hoher Präzision in der Kraftmess- und Wägetechnik.

In der Forschung gibt es ein großes Potenzial der Dynamikverbesserung durch schnellere mechatronische Systeme und Störunterdrückung bei beiden Wägeprinzipien. Einen Ansatz zur Steigerung der Leistungsfähigkeit bietet der Aufbau volldigitaler Wägesysteme, in denen die parallele Anregung und Überwachung aller relevanten Baugruppen und Zustände möglich wird [WGA13][BS91][Pf96].

In dieser Arbeit stehen die Möglichkeiten der digitalen Informationsverarbeitung von EMK-Waagen im Vordergrund. Deswegen wurde unter anderem eine gründliche Recherche im Bereich der Signalverarbeitung für diese Wägesysteme durchgeführt.

Die Waagen nach dem EMK-Prinzip sind typische mechatronische Systeme. Um sie auch in hochdynamischen Anwendungen verwenden zu können, müssen alle Komponenten, insbesondere die Mechanik, die Signalverarbeitung sowie die Regelung hinsichtlich ihrer Geschwindigkeit optimiert werden [Ba15][Kr04].

Die Abbildung 2.3 stellt den schematischen Aufbau einer EMK-Waage dar. Das EMK-Messprinzip besteht darin, ein Gleichgewicht der zu messenden Kraft mit einer gleichgroßen elektromagnetisch erzeugten Gegenkraft zu erreichen.

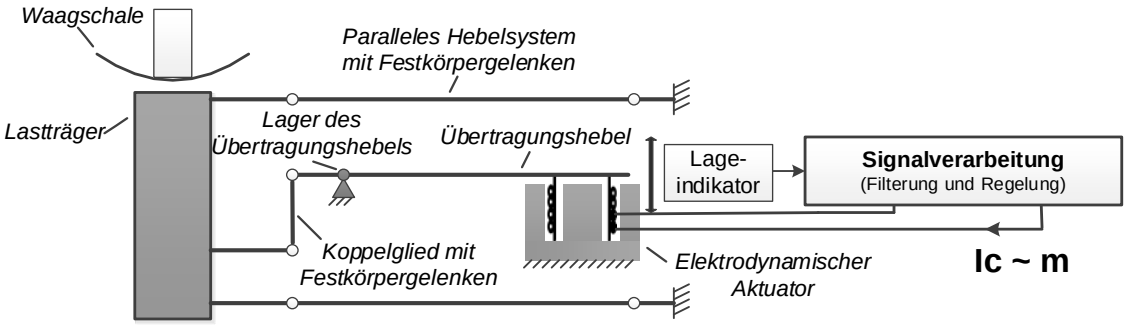


Abbildung 2.3: Schematischer Aufbau einer EMK-Waage

Wenn die Waagschale befüllt ist, wird eine Kraft erzeugt. Dabei erfolgt ein Absenken im parallelen Hebelsystem durch eine entsprechende Führung. Diese Bewegung wird durch den Übertragungshebel in einem Verhältnis übertragen. Der Lageindikator erfasst die Verschiebung des Übertragungshebels. Am Ende des Übertragungshebels wird die Vergleichskraft durch eine Magnetspule in einem permanent-magnetischen Feld (durch einen elektrodynamischen Aktuator) erzeugt. Diese Kraft wird durch die Signalverarbeitung so geregelt, dass die aus diesen Kräften resultierenden Momente sich im eingeschwungenen Zustand kompensieren. Im Resultat wird der Hebel durch die Regelung in seiner horizontalen Ruhelage gehalten. Für die Hebelregelung verwendet man einen sehr empfindlichen Lageindikator (auch Nullindikator genannt) aus einer Leuchtdioden-Schlitzblenden-Photodioden-Anordnung, der die vertikale Position des Hebels erfasst und diese in einen Photostrom umwandelt. Dieser Strom ist der Eingang eines meistens in Analogelektronik aufgebauten Reglers. Der Ausgang des Reglers ist eine Spannung, die den Strom durch

die Spule als Stellglied steuert. Der durch Spule fließende Strom (*Ic*) wird im eingeschwungenen Zustand als Maß für die Kraft beziehungsweise die Masse gemessen. Dieser wird nach einer analog-digitalen Umwandlung und einer digitalen Filterung angezeigt [MS89][Do08][BHH95].

In der Vergangenheit wurde bereits eine Vielzahl von Arbeiten zur digitalen Informationsverarbeitung für dynamische Waagen publiziert. Dazu gehören unterschiedliche Untersuchungen zur Verbesserung und Weiterentwicklung von Filter- und Regelungsalgorithmen.

Die Arbeit von Krause „Dynamische Wägetechnik“ [Kr04] stellt eine vergleichende Untersuchung von verschiedenen Wägeprinzipien dar, die in Kontrollwagen genutzt werden. Es wurden Federwaagen, Dehnungsmessstreifen- und Kreiselwägezellen, interferometrische und kapazitive Wägezellen und Wägezellen mit elektrodynamischer Kompensation analysiert. In der Signalverarbeitung der betrachteten Wägeprinzipien wurde auf die Regelung (beziehungsweise Störungskompensationsverfahren mit PID-Reglern (s. auch [Ng10][MS89])) und die Messsignalfilterung eingegangen. Im Fokus dieser Dissertation steht die Problematik der schnellen Massenbestimmung, die beispielweise in industriellen Fertigungsprozessen auftritt. Es wurde nachgewiesen, dass ein FIR-Filter sowohl die Beeinflussung des Frequenzganges als auch die Erhöhung der Amplitudenauflösung ermöglicht.

Eine weitere Arbeit von Dontsova „Digitale Signalverarbeitung in der dynamischen Wägetechnik“ diente hauptsächlich zur Untersuchung und Realisierung von einer geeigneten Filterung vor der Anzeige von Messwerten. Es wurden Algorithmen zur adaptiven Filterung in Form von mehreren in Reihe geschalteten (kaskadierten) Mittelwertbildnern vorgeschlagen. Als Realisierungsplattform dient der Mikrocontroller MB90F553A der Firma Fujitsu. Dieser wird hauptsächlich für die Steuerung des Messablaufes, zur Einstellung von Filterkoeffizienten, zur Korrektur von Umwelteinflüssen, sowie zur Kommunikation mit einem PC verwendet. [Do08]

In der Dissertation von Pfeiffer [Pf96] wurde die digitale Signalverarbeitung für EMK-Waagen basierend auf dem Entwurf eines digitalen integrierten Regler- und Filterkonzepts dargestellt. Der Schwerpunkt der Arbeit liegt auf dem H_∞-Regler mit integriertem FIR-Filter. Der Regler/Filter wurde in diesem Aufbau mit einem Signalprozessor realisiert. Dieses Konzept bietet Verbesserungen bezüglich der erreichbaren Einschwingzeit und der dynamischen Eigenschaften der Präzisionswaagen, und die Reduzierung von entstehenden Schwankungen in der Anzeige [PS94].

In [MSch89] wurde eine mögliche Realisierung eines digitalen PID-Reglers beschrieben. Der integrale Anteil des PID-Reglers wird zur Schätzung des Messwertes verwendet und der proportionale und der differenzierende Anteile werden erst nach dieser Stufe zu dem Ausgangsignal addiert und in einen Strom gewandelt. Dieses Konzept erfordert eine sehr

hohe Auflösung und Driftarmut des DA-Wandlers. Für die Filterung der Messwerte wurden zwei gleitende Mittelwertfilter und ein Kalman Filter vorgeschlagen. Es besteht die Möglichkeit abhängig vom aktuellen Zustand der Waage zwischen den beiden Algorithmen umzuschalten.

Die Publikation [BHH95] stellt einen digitalen Regler mit integriertem Filterkonzept vor. Basierend auf der Methode „Linear Quadratic Gaussean“ wurden ein linearer Regelalgorithmus und ein Beobachter in Form eines Kalman Filters entworfen.

In [BS91] wurden mehrere Lösungen der digitalen Regelung für elektromechanische Kompensationswaagen dargestellt. Es wurde vorgeschlagen, den digitalen Regler als adaptiven Regler auszulegen, unter Verwendung geeigneter Beobachtungen und Schätzungen zur Rekonstruktion der Zustandsgrößen, Modellparameter und Prozesskoeffizienten.

Die ständige Verbesserung und Erhöhung der Komplexität der eingesetzten Mess-, Überwachungs-, Steuerungs- und Regelungsalgorithmen steigert den Bedarf an zuverlässigen, leistungsfähigen und langzeitstabilen Messdatenerfassungssystemen. Die störungsfreie Zusammenarbeit unterschiedlicher Prozesse der Messkette benötigt häufig den Einsatz von Mehrrechnersystemen, die zusätzliche Anforderungen an Kommunikation und Synchronisierung erzeugen [MAFA09][WW10][ZKGA13].

Im Abschnitt 2.4.3 werden unterschiedliche Implementierungsplattformen detaillierter dargestellt. Es wird dabei auch begründet, warum sich diese Arbeit auf die Verwendung eines FPGA-basierten Eingebetteten Systems für die Informationsverarbeitung in Messsystemen konzentriert.

2.3.2 Gesichtspunkt der Eichfähigkeit und anderer Empfehlungen für die digitale Informationsverarbeitung in der Wägetechnik

In diesem Teil wird ein Überblick zu speziellen Anforderungen und Bedingungen für ein Informationsverarbeitungssystem in einem eichfähigen Messsystem gegeben. Dazu werden die Regelungen der PTB und der WELMEC am Beispiel der eichfähigen Wägetechnik näher betrachtet und erläutert.

Seitdem die Mikroprozessortechnologie in der Messtechnik Einzug gehalten hat, bekommt die Software immer größere Bedeutung. 1995 wurden EU-einheitlichen Software-Anforderungen für eichfähige, PC-gestützte Wägetechnik durch die WELMEC zusammengefasst. Diese gelten bezüglich der Struktur, dem Niveau des Manipulationsschutzes, der Identifikation und der Dokumentation der Software [WEL14][PTB06].

Die allgemeinen Regelungen im aktuellen „Softwareleitfaden“ WELMEC 7.2 [WEL04] unterteilen alle Systeme in unterschiedliche Risikoklassen, wobei sich diese aus drei Stufen zusammensetzen: Stufe für den Softwareschutz, für die Softwareprüfung und für die

Softwarekonformität. Entsprechend der Risikoklasse ergeben sich verschiedene Anforderungen an die Software und demzufolge an deren Dokumentation. Die Systeme mit „*eingebetteter Software in einem Messgerät mit zweckgebundener Hard- und Software*“ werden als ***Typ P*** bezeichnet. Die Dokumentation zu dem entwickelten Informationsverarbeitungssystem muss die folgenden Punkte enthalten:

- Beschreibung der festen beziehungsweise rechtlich relevanten Software,
- Beschreibung der Messgenauigkeit der implementierten Messalgorithmen,
- Beschreibung der Benutzerschnittstellen,
- Beschreibung der Realisierung zur Softwareidentifikation,
- Darstellung der Systemhardware und einer Systemübersicht z.B. in Form eines Blockschaltbildes, wenn dies nicht im Benutzerhandbuch erfolgt,
- Benutzerhandbuch,
- für Messgeräte der Risikoklasse E (z.B. ein größerer Teil der Waagen) muss zusätzlich der gesamte Quellcode eingereicht werden.

Jeder der genannten Punkte muss detailliert bezüglich seines Zusammenhanges und seiner Struktur in der Software beschrieben werden. Änderungen an der Software können nur mit Genehmigung der angegebenen Stelle (z.B. bei der PTB) durchgeführt wurde. Jede Änderung, die an der rechtlich relevanten Software durchgeführt wurde, erfordert eine neue Softwareidentifikation und Zulassung.

Ein weiterer wichtiger Leitfaden ist der „*Leitfaden zu den allgemeinen und verwaltungstechnischen Aspekten des freiwilligen Systems zur modularen Bewertung von Messgeräten*“ WELMEC 8.8 [WEL12]. Dieser beschreibt die unterschiedlichen Teile der Konformitätsbewertung (in Deutschland liegt die Verantwortung bei der PTB). Dabei wird eine systematische Untersuchung durchgeführt, in wieweit ein Produkt, ein Prozess oder eine Dienstleistung die festgelegten Anforderungen erfüllt (s. auch DIN EN 45020:2006 [DN06]).

Im „Leitfaden zur Softwareprüfung“ WELMEC 2.3 sind die Anforderungen an die Software eines wägetechnischen Systems beschrieben, nach denen geprüft wird. Die Tabelle 2.1 stellt diese zusammenfassend dar.

Der WELMEC 2.3 wird durch den „*Leitfaden zum Modulkonzept und zur Prüfung von PCs und anderen digitalen Zusatzeinrichtungen*“ WELMEC 2.5 erweitert. In diesem geht es um die Speicherung, Weiterverarbeitung und Identifikation von rechtlich relevanten Messdaten und Messergebnissen außerhalb des Messgerätes. [WEL14]

	Anforderungen
Softwareschutz	• Software muss gegen beabsichtigte, unbeabsichtigte und zufällige Manipulation geschützt sein. Eine Manipulation mit gängigen Softwareprogrammen muss dabei ausgeschlossen werden. • Es muss sichergestellt werden, dass Variablen innerhalb der Software nicht verändert werden können.
Identifizierung	• Software muss jederzeit identifizierbar sein. Dazu ist es erforderlich, eine Prüfsumme (engl. *checksum*) oder eine andere Form der Signatur zu verwenden. Mithilfe dieser wird überprüft, ob die aktuell verwendete Software mit der bei der Zulassung geprüften Software übereinstimmt
Softwareschnittstellen	• Schnittstellen innerhalb der Software, sowie die externen Schnittstellen müssen geschützt werden. Das Umgehen des vorhandenen Schutzes durch den Benutzer stellt dabei eine kriminelle Tat dar. Eine genaue Beschreibung, wie ein solcher Schutz aussehen könnte, wird nicht gegeben.
Softwaredokumentation	• Darlegung aller rechtlich relevanten Softwareteile, Parameter und aller Funktionen an denen diese beteiligt sind. • Sämtliche Befehle für das Messgerät über die gesicherte Schnittstelle müssen angegeben werden. • Genaue Beschreibung der Sicherheitsfunktionen und deren Funktionsweise. Außerdem muss erklärt werden, wie die Softwareidentifikation ausgeführt werden kann. • Schriftliche Erklärung, dass die angegebenen Informationen vollständig sind. • Schriftliche Erklärung, dass der Standard EN 45501:1992/AC 1993 eingehalten wurde.
Zusätzlich	• Es darf nur erlaubte Software geladen werden. Die Funktionsweise dieser Schutzfunktion muss in der Dokumentation enthalten sein. Außerdem muss die Software innerhalb des Gerätes in einem nicht flüchtigen Speicher so gesichert sein, dass sie weder verändert noch so manipuliert werden kann, dass es nicht mehr funktionsfähig ist. • Es soll die Möglichkeit zur Überprüfung der Authentifizierung und Integrität der geladenen Software bei der Zulassung bestehen. Trotzdem muss es weiterhin möglich sein, dies auch durch den Nutzer zu verifizieren. • Das Laden der Software muss nachvollziehbar sein. Es muss ein Logger (Registriereinrichtung) geben, z.B. ein Zähler, der die Anzahl von Software-Downloads verfolgt. Diese müssen wiederum gegen Manipulation geschützt sein, wobei beim Erreichen der maximalen Anzahl an Ladevorgängen keine weiteren Software-Downloads möglich sind. Es muss sichergestellt werden, dass die Software nur geladen werden darf, wenn der Benutzer dies durchführen will. Die Funktionsweise ist wiederum in der Dokumentation darzulegen.

Tabelle 2.1: Spezielle Software-Anforderungen nach WELMEC 2.3

Die oben beschriebenen Regelungen gelten großteils für die entwickelte Software in Eingebetteten Systemen (außer WELMEC 2.5). Die Informationsverarbeitungssysteme mit FPGA-basierten Plattformen werden als Hardware-Software-Systeme interpretiert. Bei der WELMEC und der PTB sind keine speziellen Regelungen für diese vorhanden. Für FPGAs existieren bereits unterschiedliche Normen und Standards anderer Institutionen, wie z.B. die Norm DO-178B/254, Standard IEC 61508 [CTB14][CG13]. Diese beschreiben die Sicherheits- und Zertifizierungsanforderungen bei der Entwicklung von FPGAs für allgemeine Anwendungen.

Basierend auf den in diesem Abschnitt dargestellten Regelungen, Normen und Standards werden in der Dissertation weitergehende eigene Lösungsansätze für die FPGA-basierten Eingebetteten Systeme in der Messtechnik entwickelt und präsentiert.

2.4 Entwicklungsprozesse für die Informationsverarbeitung mit Eingebetteten Systemen

Die komplexe digitale Informationsverarbeitung in der Messtechnik wird zunehmend durch die Notwendigkeit des Arbeitens mehrerer Gruppen aus unterschiedlichen Fachgebieten (z.B. Messtechnik, Regelungstechnik, Informationstechnik) erschwert. Der kritische Punkt ist in diesem Kontext der Ablauf eines gemeinsamen systematischen Vorgehens im Rahmen eines definierten Prozesses und eines weitgehend einheitlichen Modellierungs- und Testvorgehens (einschließlich der Modellierungssprachen). Das reicht bis zur unterschiedlichen Interpretation von gemeinsam verwendeten Fachbegriffen [Ar12]. Aus diesem Grund existieren hohe Anforderungen an die koordinatorische, gemeinsame und systematische Entwicklungsarbeit. Es ist ein Konzept notwendig, in dem die einzelnen Teilsysteme und -aufgaben identifiziert und festgelegt werden und ihre komplexe Zusammenwirkung beschrieben wird.

In den zurzeit eingesetzten Vorgehensweisen wird die Entwicklung eingebetteter messtechnischer Systeme von mechanischen und elektronischen Gesichtspunkten getrieben. Besonderheiten von Entwicklungsprozessen für Eingebettete Systeme bestehen u.a. darin, dass eine abgestimmte Zusammenarbeit von Subsystemen unterschiedlicher Disziplinen (z.B. Mechanik, Messelektronik, Hard- und Software) notwendig ist, wobei dieses eine große Herausforderung darstellt [LR05]. Eingebettete Systeme übernehmen in der messtechnischen Anwendung die Signalverarbeitung, Steuerung und Regelung. Auf Grund des Bedarfs an ständig kürzeren Messzeiten müssen die Informationsverarbeitungslösungen häufig strikten Zeitanforderungen genügen.

Die maximale Reaktionszeit des Systems ist nicht nur durch die Leistungsfähigkeit der Hard- oder Software, sondern auch durch die Eigenschaften des gesteuerten technischen Prozesses bestimmt [BST10]. Ein weiterer wesentlicher Unterschied gegenüber der Ent-

wicklung von reiner PC-Software ist die Möglichkeit leistungsunterschiedlicher Hardware-Zielplattformen (nicht nur auf Mikrocontroller und DSPs beschränkt), auf deren Basis die notwendigen Algorithmen realisiert werden können. Dabei wird die Software nicht direkt auf dem Zielsystem entwickelt und muss dorthin später portiert beziehungsweise ein spezieller Code generiert werden. Zwischen den Entwicklungs- und Zielplattformen sind einige Unterschiede (z.B. Zeitverhalten, Hardware-Ressourcen) vorhanden. Aufgrund dessen wird das Zielsystem auf einem Rechner häufig emuliert [Be05] oder die Algorithmen in einer implementierbaren Form simuliert [G08].

Ein weiterer Punkt bei der Entwicklung von Eingebetteten Systemen ist die systematische Beschreibung aller Entwurfsetappen der Teilaufgaben und die Zusammenführung in ein funktionierendes Gesamtsystem aus Eingebettetem und Einbettendem System. Es existiert kein standardisierter Prozess zur Entwicklung Eingebetteter Systeme, der die Spezifika der Messtechnik und insbesondere der Wägetechnik vollständig berücksichtig. Jedoch sind effiziente Entwicklungsprozesse notwendig, die auf die technischen Besonderheiten und die Anwendungsdomäne abgestimmt werden. Außerdem müssen diese Prozesse qualitativ hochwertig, nachvollziehbar und wiederholbar realisiert werden. Der spezielle Entwicklungsprozess für Eingebettete Systeme in messtechnischen Systemen unterliegt damit besonderen Bedingungen und Herausforderungen, wie auch in anderen sicherheitskritischen Systemen und Systemen mit harten Echtzeitbedingungen [Be05][BST10]. Diese sind das Erreichen einer hohen Zuverlässigkeit, Robustheit, Manipulationssicherheit, kurzer Latenzzeiten und möglichst geringer Kosten des zu entwickelnden Produkts [HSD09]. Vor allem in der Wägetechnik wird ein Nachweis verlangt, dass ein Produkt die entsprechenden Gesetze der Eichbehörde, Normen, Richtlinien und Standards (mehr im Abschnitt 2.3.2) erfüllt. Diese stellen Anforderungen an die Messgeräte, deren Nachweis und Gewährleistung während Entwicklung und Betrieb dar. Aus diesem Grund steht die Herausforderung, dass die verwendeten Entwicklungsmethoden und –werkzeuge in einem Zertifizierungsprozess akzeptiert werden müssen, selbst wenn noch keine normativen Regeln zu diesen vorliegen, diese aber mit einer zu erwartenden Wahrscheinlichkeit gesetzlich geregelt werden. Als Zertifizierung [ESB07] wird ein Systemaudit bezeichnet, das von akkreditierten Organisationen durchgeführt wird. Hierbei wird die Vergabe eines Zertifikates angestrebt, das dem auditierten Produkt einen Qualitätsstandard bescheinigt.

Neben den nationalen Eichbehörden verlangen auch andere Organisationen zertifizierte Prozesse [LR05], z.B. FDA (s. Abschnitt 2.2). Das heißt, ein Prozess, der diese Forderungen erfüllt, ist für diesen Zweck zertifizierbar. Unter „zertifizierbar“ wird in dieser Arbeit verstanden, dass der Prozess mit hoher Wahrscheinlichkeit zertifiziert wird. Zurzeit werden solche Prozesse in der messtechnischen Industrie entsprechend der vorhandenen Literatur nicht oder nur rudimentär eingesetzt [HV-H02][TNRW01], insbesondere

ist die Integration von FPGA- und ASIC-Lösungen nicht enthalten. Dieses gilt mit einigen Ausnahmen für weitere Anwendungen von Eingebetteten Systemen, wie beispielweise der Standard AUTOSAR (AUTomotive Open System Architecture) in der Automobilindustrie [AUT15]. Dabei wird eine Softwarearchitektur für Steuergeräte definiert. In [AUT15: *Technical Overview*] steht:

> *„AUTOSAR enables configuration process optimization (e.g. partitioning and resource usage) and where necessary, to allow local optimization if required to meet the runtime requirements of specific devices and hardware constraints".*

Der Standard umfasst die Bereiche Architektur, Methodik und Beschreibung von Schnittstellen, um eine effiziente Softwareentwicklung zu ermöglichen. Unter anderem resultieren bei AUTOSAR als Vorteile die Wiederverwendung von Softwaremodulen, die Vereinfachung bei der Systemintegration und die Reduzierung der ausufernden Variationsvielfalt.

Im Weiteren wird der Stand der Technik zu Vorgehensweisen bei der Entwicklung eingebetteter messtechnischer Systeme, zu Prinzipien und Methoden des modellbasierten Entwurfs, zum Testen auf Modellniveau, zur Implementierung von komplexen Algorithmen der Informationsverarbeitung sowie zur Integration ins Gesamtsystem gegeben. Im Einzelnen werden die speziellen Entwicklungsprozesse präsentiert, die für die Realisierung eines Informationsverarbeitungssystems der eichfähigen Messtechnik verwendet werden können. Darauf basierend werden die Ziele des eigenen Konzepts dargestellt.

2.4.1 Vorgehensweisen bei der Entwicklung

Die Wissenschaft beschäftigt sich seit die Zeit mit der Klassifizierung und der Beschreibung von systematischen Vorgehensweisen zur Entwicklung von komplexen Systemen. Heute besteht die Erkenntnis, dass auf Grund der Vielzahl von Einflussfaktoren kein allgemein gültiges Vorgehensmodell verwendet werden kann. Dies bedeutet, dass ein eigenes Prozessmodell für das Vorgehen bei der Entwicklung eines konkreten Systems aus mehreren existierenden gewählt oder speziell entwickelt werden muss. Die bekannten Vorgehensmodelle (meist standardisierte Prozessmodelle) und Technologien zu den Entwicklungsprozessen gehören zu den relevanten methodischen Grundlagen des Projektmanagements und der Prozessoptimierung [BK08][B98]. Solche Prozesse sollen sinnvollerweise nach [R07][FFH02] werkzeugbasiert ablaufen.

Es gibt eine Vielzahl von Methoden, Techniken und Werkzeugen aus Engineering-Prozessen, die für die Entwicklung von speziellen Eingebetteten Systemen angewendet werden können. Die existierenden Vorgehensmodelle können in Familien eingeordnet werden. In [BK08] sind die vier wichtigsten Familien, d.h. sequenzielle, wiederholende, prototypische sowie wiederverwendungsorientierte Vorgehensmodelle dargestellt. Die häu-

fig verwendeten Vorgehensmodelle, die aus den Bereichen Projektmanagement und Software Engineering bekannt sind und zur Entwicklung von speziellen Eingebetteten Systemen angewendet werden können, basieren auf Phasenmodellen (sequentielles Vorgehen) mit Top-Down und Bottom-Up-Entwürfen. Im Top-Down-Entwurf werden zuerst die Ziele und die wesentlichen Anforderungen zusammengestellt und darauf aufbauend schrittweise informale und formale simulierbare Funktionsbeschreibungen entwickelt. Im Bottom-Up-Entwurf läuft die Entwicklung in entgegengesetzter Richtung. Aus Teilkomponenten mit bekannten Funktionen werden größere Funktionseinheiten zusammengesetzt [K11].

Ein verbreitetes Phasenvorgehensmodell ist das V-Modell von Boehm [B79]. Der wesentliche Unterschied des V-Modells zu anderen Vertretern der sequenziellen Modelle (z.B. zum Wasserfallmodell) ist die Integration projektbegleitender Tätigkeiten, insbesondere der Qualitätssicherung und des Projektmanagements und die frühzeitige Behandlung von Testaktivitäten. [HSD09][LL10]

Die klassische Darstellung des V-Modells (Abb. 2.4) beschreibt die Phasen der Systementwicklung vom Allgemeinen zum Speziellen auf der linken Seite nach dem Top-Down-Prinzip.

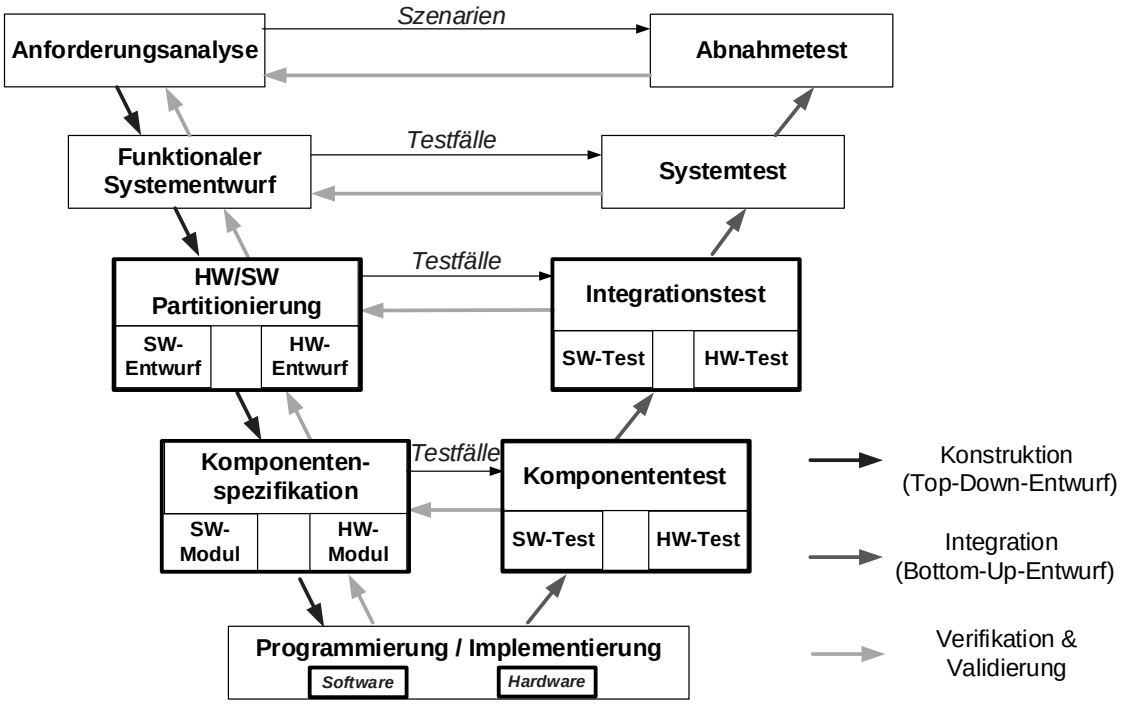


Abbildung 2.4: Klassische Darstellung des V-Modells für Eingebettete Systeme

In der ersten Phase wird die Anforderungsanalyse durchgeführt, wobei die wichtigen Eigenschaften des entwickelnden oder zu wartenden Systems festgelegt werden. Die Dokumente Lastenheft und Pflichtenheft der Anforderungsanalyse sind genormt. Für die di-

rekte Umsetzung der so beschriebenen Anforderungen werden auch andere Zwischenbeschreibungen (z.B. Use Cases) benutzt [RSG01]. Weiter wird im Systementwurf eine Gesamtlösung und im Komponentenentwurf eine detaillierte technische Lösung entwickelt, die überprüfbar den Anforderungen entsprechen. Eine Grundannahme des existierenden erweiterten V-Modells für Eingebettete Systeme ist die Unterscheidung zwischen System- und Hardware/Software-Ebenen (Hardware/Software-Codesign [G08][GM07]), sowie zwischen Analyse- und Entwurfsaktivitäten auf jeder Ebene. Hier wird eine Partitionierung des Systems in Software- und Hardware-Einheiten in frühen Phasen des Systementwurfs durchgeführt [Be05]. Als Folge dessen entstehen Teile im Prozess, die in Hardware realisiert werden, und solche, die als Software-Programme ablaufen. Bei der Partitionierung werden die Schnittstellen zwischen Hardware und Software definiert. Für die Partitionierung in analoge, digitale und Software-Teile existieren unterschiedliche Vorgehensweisen [Be05]. Nach der Partitionierung beginnt die letzte Phase des Top-Down-Entwurfs: Modulrealisierung mittels Programmierung oder Hardwarerealisierung. In diesem Prozess wird der Entwurf in Anweisungen einer Programmiersprache und Hardware Description Language (HDL) umgewandelt (auch Codegenerierung genannt) [LR05][G08].

Die rechte Seite stellt die Phasen der Integration der realisierten Teilmodule in das Informationsverarbeitungssystem dar, zusammen mit den zugehörigen Testaktivitäten nach dem Bottom-Up-Prinzip. Hierbei erfolgt die Entwicklung der Testfälle bereits beim Entwurf (linke Seite des V-Modells). Der Bottom-Up-Entwurf enthält die Phasen der Erzeugung von ausführbaren Teilsystemen beziehungsweise dem Gesamtsystem aus dem Code, und der Integration des ausführbaren Systems in die Umgebung des Anwenders beziehungsweise des Einbettenden Systems [LL10].

Für die Verlässlichkeit des Entwurfsergebnisses jeder Phase sind Verifikation und Validierung sehr wichtige Funktionen (mehr dazu im Abschnitt 2.4.4). Die Verifikation ist ein Prozess zur Beurteilung eines Systems oder einer Komponente, der feststellen soll, ob die Ergebnisse einer entsprechenden Entwicklungsphase den Vorgaben in dieser Phase entsprechen. Die Validierung (oft auch eng. „Validation“) umfasst alle Aktivitäten mit dem Ziel festzustellen, ob der Einsatzzweck oder die Benutzererwartungen erfüllt werden [SZ03]:

> *„Die Funktionsvalidierung ist demnach eine Prüfung, ob die Spezifikation die Benutzeranforderungen erfüllt, ob überhaupt die Benutzerakzeptanz durch eine Funktion erreicht wird“*.

In [LR05] sind weitere V-basierte Vorgehensmodelle dargestellt, die u.a. für die Entwicklung eingebetteter Software erstellt wurden. Eines davon ist das V-Modell XT.

Seit 2004 ist das V-Modell XT („extreme Tailoring“) als ein generischer Vorgehensstandard für Projekte in der Wissenschaft bekannt [HH08]. Es unterstützt drei Projekttypen:

Systementwicklung eines Auftraggebers, Systementwicklung eines Auftragnehmers und Einführung und Pflege eines organisationsspezifischen Vorgehensmodells. Unter der Systementwicklung werden dabei Entwürfe eines Systems und zusätzlicher Unterstützungssysteme (z.B. Test- und Wartungssystemen) verstanden. Die Einheiten des Systems werden auch hierarchisch in Segmente, Hardware-Einheiten und Komponenten, Software-Einheiten und Komponenten, Externe Einheiten, Hardware- und Software-Module gegliedert [HH08]. Ähnlich wie im klassischen V-Modell werden zunächst die Anforderungen aus den übergeordneten Systemelementen übernommen, die weitere hierarchische Zerlegung des Systems entworfen, die Systemspezifikation realisiert und schließlich die Anforderungen der nächsten Ebene der Systemelemente zugeordnet. Die weitere Realisierung von Hardware- und Software-Modulen erfolgt nach dem Botton-Up-Entwurf. Das V-Modell XT unterstützt drei Methoden der Systementwicklung: inkrementelle, komponentenbasierte und prototypische [LL10]. In [RN05] sind die wichtigsten Vorteile des Modells vorgestellt, beispielsweise Minimierung von Projektrisiken, Verbesserung und Gewährleistung der Qualität der Entwicklungsergebnisse.

Ein Vorteil ist unter anderem, dass das allgemeine V-Modell sich für eine konkrete Anwendung in weitem Rahmen anpassen und erweitern lässt. In [HSD09][LR05][Be05] sind unterschiedliche Erweiterungen für die Anwendung bei der Entwicklung komplexer Eingebetteter Systeme beschrieben. Obwohl in den letzten Jahren eine Menge von Verfeinerungen und Anpassungen an bestimmte Prozesse diskutiert wurden, bleiben im V-Modell der Mangel an der Testbarkeit in früheren Entwicklungsphasen und die undeutliche Trennung in Test-, Debug- und Änderungsaktivitäten [LL10][SRWL11].

Das im Jahr 1993 von Herzlich definierte W-Modell [GT02], das eine Erweiterung des V-Modells darstellt, sieht statische Testaktivitäten in jedem Schritt vor: von der Anforderungsspezifikation über Spezifikation und Design bis zur Implementierung (Abb. 2.5).

Weiterhin enthält dieses Modell dynamische Testmöglichkeiten vom Modul über Modulintegration, Systemintegration und Systemabnahme. Es entstehen zwei parallele V-Modelle: Das erste enthält die Etappen des bekannten V-Modells ohne Testaktivitäten und das zweite beschreibt die Testaktivitäten. Dabei werden links die Testschritte zur Anforderungsspezifikation, zur funktionalen Systemspezifikation und zum Systementwurf durchgeführt. Dieses ermöglicht die entwurfsbegleitende Validierung jedes Schrittes, zur frühen und systematischen Erkennung und Korrektur von Fehlern. Die rechte Seite des zweiten V-Modells enthält die Testdurchführung nach der Implementierung und in den verschiedenen Integrationsstufen. In diesem Zweig werden Komponententest, Integrations- und Systemtest und schließlich Abnahmetest stattfinden.

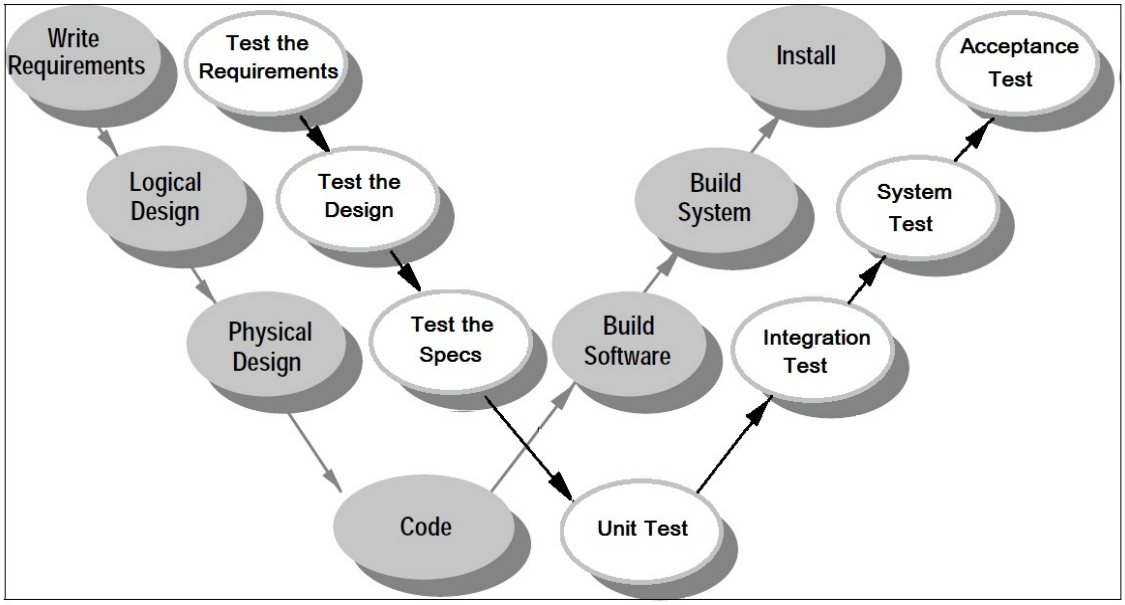


Abbildung 2.5: W-Modell von Herzlich, aus [GT02]

In der Arbeit von Müller [Mü12] wurde dieses Vorgehen ansatzweise als Entwicklungsmodell für Eingebettete Systeme mit rekonfigurierbarer Hardware (FPGAs) vorgeschlagen. Darüber hinaus wurde eine Erweiterung im unteren Teil des W-Modell-Prozesses (von funktionaler Systemspezifikation zur Systemintegration) für das Codesign von eingebetteter Plattform, Hardware-Elementen und Software-Funktionen dargestellt. In diesem Punkt schließt sich der Hardware-Software-Partitionierungsprozess an, in dem die Entwürfe und die dazugehörigen Tests nicht unabhängig voneinander ablaufen. Das Codesign wird durch Simulation, Co-Simulation und Emulation unterstützt. Nach der Implementierung von Hardware- und Software-Modulen schließen sich die Schritte der Integration und der Integrationstest an. Es fehlen allerdings die Berücksichtigung des Einbettenden Systems und dessen Modelle.

Eine andere Variante des W-Modells für reine Softwareprojekte wurde von Spillner 2002 beschrieben [SRWL11]. In diesem Modell (Abb. 2.6) werden Testaktivitäten (Planung, Analyse und Design) parallel zu der Top-Down–Entwicklung im V-Modell bearbeitet.

Die rechte Seite des W-Modells enthält Testdurchführung, Debugging und Veränderungen zur Fehlerkorrektur im Bottom-Up-Verfahren. Der Hauptvorteil dieses W-Modells besteht darin, dass die Tester ab dem Start des Projektes an der Entwicklung beteiligt sind. Parallel zur Anforderungsdefinition werden beispielweise die entsprechenden Abnahmetestfälle spezifiziert und gefolgt vom nächsten Entwicklungsschritt, in dem geprüft wird, ob die Anforderungen die nötige Präzision erfüllen. Der Testprozess ist eng mit dem entsprechenden Entwicklungsprozess verzahnt, aber dennoch ein eigenständiger Prozess. Es wird möglich, unpräzise Angaben zu erkennen und zu präzisieren, da zur Erstellung der Testfälle genaue Angaben erforderlich sind. Außerdem liefern die erstellten Testfälle eine eher formale Spezifikation für die weiteren Entwicklungsschritte. Die frühzeitige

Spezifikation der Testfälle ermöglicht auch, Anforderungen klarer und stabiler zu formulieren. Weiterhin kann die Spezifikation als Dokumentation jedes Entwurfsschrittes verwendet werden. [SRWL11]

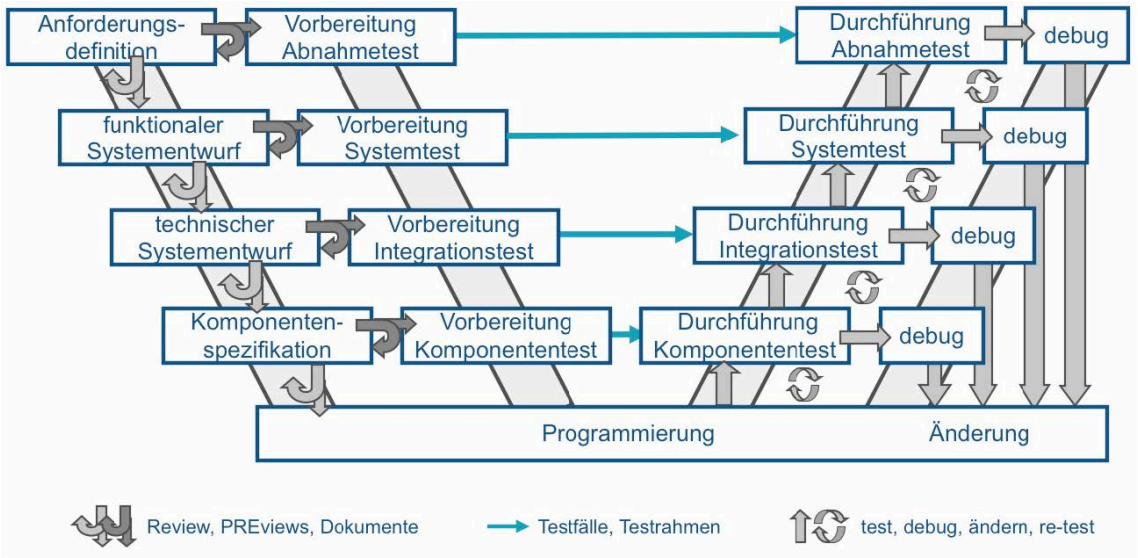


Abbildung 2.6: W-Modell von Spillner, aus [SRWL11]

Bei der Entwicklung komplexer eingebetteter Informationsverarbeitungssysteme für die messtechnische Anwendung spielen die Testaktivitäten in früheren Phasen eine große Rolle. Außerdem soll die Möglichkeit existieren, die gefundenen Fehler in den Testläufen auf der Hierarchieebene, auf der sie entstanden sind, zu korrigieren. Die Verwendung von einem W-Modell kann dadurch motiviert werden, dass es zum modellbasierten Entwurf von komplexen Systemen führt [SFM12]. Dabei wird ein virtueller Prototyp auf Basis von simulierbaren Modellen entwickelt, in dem eine ausführbare Spezifikation und modellbasierte Testgenerierung möglich sind [Mü12][Ma11][Bau09]. Die Prototypen ermöglichen den frühzeitigen Nachweis von spezifischen Produkteigenschaften. Sie helfen unklare Anforderungen genauer zu fassen und dienen zur Demonstration sowie zur Validierung von Entwürfen. Ein weiterer Vorteil liegt in der Akzeptanz bei Kunden/Endbenutzern bereits in den früheren Phasen. Deswegen wird im Weiteren das Vorgehen auf Basis der W-Modelle von Herzlich und Spillner als Rahmen eines zertifizierbaren Prozesses in der genannten Domäne benutzt werden, wobei eine Reihe von Modifikationen und Erweiterungen notwendig wird. Diese werden im Kapitel 3 beschrieben.

In der Dissertation von Farrag [Fa10] wurde ein weiteres W-Modell für das Gebiet Produktlinienentwurf entwickelt. Es enthält zwei V-Modelle angelehnt an Boehm. Eines beschreibt den Domänenentwurf und das zweite den Applikationsentwurf. Zusätzlich enthält es die Beziehungen zwischen beiden. Dieses W-Modell hat weniger Bezug zu dieser Arbeit. Es wird interessant, wenn in dem gesamten Prozess auch Produktlinienaspekte zur Betrachtung hinzugezogen werden. Einige Informationen zu Grundlagen und zum Entwurf von Produktlinien werden im Abschnitt 2.4.5 gegeben.

2.4.2 Modellbasierte Entwicklung von Eingebetteten Systemen

2.4.2.1 Modelldefinition

Zur vollständigen Modellierung eines Systems ist die Entwicklung eines Modells notwendig, anhand dessen man unterschiedliche Aspekte dieses Systems entscheiden kann und die für das konkrete System relevanten Aspekte feststellt und formal beschreibt. Aus diesem Grund ist der erste Schritt bei der Modellierung die Suche nach den für die Entwicklung relevanten Eigenschaften des Systems. Die Voraussetzung für diese Analyse ist die Suche einer geeigneten Darstellung für die gefundenen Eigenschaften des Systems. Ein nächster Schritt bei der Modellierung ist die Feststellung, in welcher Beziehung die einzelnen Modelle beziehungsweise Teilmodelle zueinander stehen. In einigen Fällen gibt es mehr oder weniger komplexe Gesetzmäßigkeiten, in anderen Fällen sind heuristische Annahmen möglich. Die heuristischen Annahmen kann man durch Beobachtungen oder Messungen erkennen [EGK08][Ja10].

Ein Modell im betrachteten Kontext besteht in der Regel aus Anweisungen, Variablen und einer Steuerung der Anweisungsreihenfolge, womit zumeist ein Algorithmus ausgedrückt wird [MA11]. Dementsprechend können bei der Betrachtung zwei Aspekte unterschieden werden: Betrachtung des Datenflusses (Abhängigkeiten von Variablenwerten) und Betrachtung des Steuerflusses [Zu06]. Die datenflussorientierten Modelle (auch aktivitätsorientiert genannt [Tei97]) werden in Form von Datenflussgraphen dargestellt und sind am besten für Systeme geeignet, die durch Dateneingabe gesteuert werden (z.B. in der digitalen Signalverarbeitung). Für den Steuerfluss werden eine deklarative Semantik (Bedingungen zur Ausführung und die Reihenfolge werden nicht explizit spezifiziert) und eine prozedurale Semantik (Reihenfolge der auszuführenden Schritte wird definiert) verwendet [B95]. Bei der expliziten Betrachtung des Steuerflusses werden kontrollflussdominante oder zustandsorientierte Modelle verwendet [Tei97][NM10]. Diese sind hier vorwiegend zur Modellierung von Steuerungsaufgaben (z.B. in reaktiven Echtzeitsystemen) geeignet. Diese Modelle repräsentieren das betrachtete System als eine Menge aus Zuständen und Zustandsübergängen (z.B. Modell des endlichen Automaten – „*finite state machine*“ FSM [Ma11]). Als ein Mittel der Modellierung werden Zustandsdiagramme eingesetzt. Ein häufig verwendetes zustandsorientiertes Modell zur Spezifikation von Eingebetteten Systemen ist das Modell des hierarchischen, nebenläufigen endlichen Automaten (engl. *hierarchical concurrent finite-state machine*, HCFSM [Tei97]). Eine Implementierung dieses Modells sind Statecharts nach Harel [Zu06][H87].

Eine weitere Klassifizierung von Modellen unterscheidet strukturorientierte, zeitorientierte (zeitbehaftete markierte Graphen) oder datenorientierte Modelle [RBGW10][G08]. Wenn eine Beschreibung von Modellteilen und deren Verbindungen im System benötigt wird, werden Blockschaltbilder benutzt, die sich auf strukturorientierte Modelle beziehen.

Als Weiteres existieren zustandsorientierte Modelle, die auf Petri-Netzen basieren und besondere Eigenschaften besitzen. Petri-Netze (nach dem Mathematiker C.A. Petri [P75]) sind ein grafisches und mathematisches Modellierungsmittel, das für Systeme aus verschiedenen Anwendungsdomänen anwendbar ist, z.B. asynchrone, verteilte, parallele, deterministische und/oder stochastische Typen von Systemen [Zi08]. Damit sind sie auch für die Informationsverarbeitung geeignet [FP91]. Als grafisches Mittel können Petri-Netze für die anschauliche Darstellung des modellierten Systems, analog zu Strukturdiagrammen, Blockdiagrammen und Netzgrafiken, benutzt werden. Es existiert eine Vielzahl von Erweiterungen und Modifikationen von Petri-Netzen für unterschiedliche Anwendungsprobleme [Re10][JBK09].

Die oben genannten Modelle beschreiben eine bestimmte Eigenschaft beziehungsweise Sichtweise eines Systems und sind leichter zu analysieren. Die Modellierungskraft ist hingegen eingeschränkt, wenn ein komplexes System beschrieben werden muss [Tei97]. Dann werden heterogene Graphenmodelle verwendet, die unterschiedliche Eigenschaften gleichzeitig darstellen können. Für die in dieser Arbeit betrachteten Hardware-Software-Systeme ist es wichtig sowohl Kontrollfluss als auch Datenfluss in einem Modell zu berücksichtigen [DLJ99][BRS13]. Die heterogenen Graphenmodelle sind Kontroll-Datenflussgraphen (engl. *control/data flow graphs* CDFGs). In [Tei97] steht:

> *„Das Modell von Kontroll-Datenflußgraphen ist nun ein heterogenes Modell, das eine Aufteilung einer Systemspezifikation in kontrollflußorientierte und datenflußorientierte Komponenten vornimmt und damit beide Aspekte in einem Modell vereinigen kann“*.

Als vernünftiger Kompromiss zwischen einer rein mathematischen Beschreibung des Informationsverarbeitungssystems und der Transformation in einen analysierbaren oder ausführbaren Code bietet sich die Spezifikation eines allgemeinen, möglichst sprach- und plattformunabhängigen Simulationsalgorithmus an [Bau09][Pa09].

2.4.2.2 Modellbasierter Entwurf und modellbasiertes Testen

Die Entwicklung von komplexen eingebetteten Informationsverarbeitungssystemen lässt sich effektiv nur unter Nutzung eines modellbasierten Entwurfsverfahrens durchführen. Einige Beispiele des modellbasierten Entwurfs eingebetteter Software findet man in [KKF08][KCF04][MAFA09]. Es werden dabei bereits frühzeitig im Entwicklungsprozess in der Beschreibungssprache des Modellierungs- und Simulationswerkzeuges ein Funktionsmodell (z.B. der Steuerungs- und Regelungssoftware) und ein Modell des umgebenden Systems (z.B. Streckenmodell) erstellt und ausführbare Teile im Verbund simuliert [Ja10][Co04].

Unter einem modellbasierten Entwurf wird die Überführung der funktionellen Anforderungen für ein Zielsystem in eine diese Anforderungen erfüllende Realisierung verstanden [Mü12]. Das Validieren der Realisierungsstruktur gegenüber den Anforderungen wird dabei ergänzt durch das Validieren des Modells gegenüber den Anforderungen und der Realisierungsstruktur gegenüber dem Modell. Es werden zum Teil auf den unterschiedlichen Abstraktionsebenen beziehungsweise den verschiedenen Aspekten des Systems unterschiedliche Modellierungsmitteln (z.B. Sprachen) genutzt, für die Zusammenhänge existieren. Die Systemanforderungen, die in der Anforderungsanalyse untersucht werden, bilden gleichzeitig die Grundlage zur Festlegung des erforderlichen Abstraktionsniveaus [NM10]. Bei komplexen Systemen werden meist hierarchische Modelle genutzt, wobei der Abstraktionsgrad zur Realisierung hin abnimmt [EGK08].

Die Abbildung 2.7 stellt einen prinzipiellen Ablauf des modellbasierten Entwurfs und der Abstraktionsebenen dar. Die erste Ebene ist eine Beschreibung in Form eines *Computation Independent Model (CIM)* [BST10].

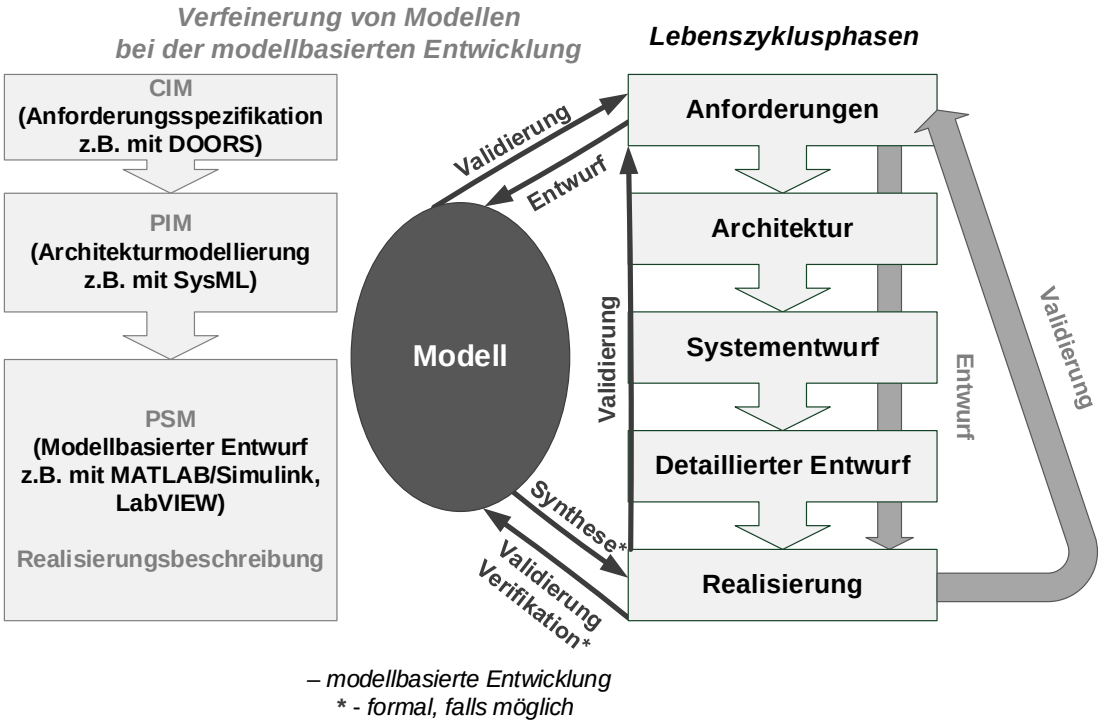


Abbildung 2.7: Prinzip des modellbasierten Entwurfs

Eine Voraussetzung für den effektiven modellbasierten Entwurf ist die Verfügbarkeit von ausführbaren Modellen. Hierdurch kann der Entwurf simulativ analysiert und gegen die Systemanforderungen validiert werden. Der modellbasierte Entwurf unterstützt die wichtigsten Ermittlungen in den früheren Phasen und die Bewertung von unterschiedlichen Realisierungsmöglichkeiten auf verschiedenen Abstraktionsebenen [Ma11]. Man unterscheidet plattformunabhängige (*Platform Independet Model - PIM*) und plattformspezifische Modellierung (*Platform Specific Model - PSM*) [BST10]. Modellunsicherheiten

und -fehler können beschleunigt aufgefunden, dynamische Ressourcenauslastungen bemessen und die Einhaltung von Randbedingungen überprüft werden [GDMF12][Mü12]. Zusätzlich kann die Realisierungsstruktur gegenüber dem Modell formal verifiziert werden, wenn für die Modellierungssprache geeignete Techniken entwickelt wurden. In diesem Zusammenhang ist zu beachten, dass eine Verifikation für alle Beschreibungsmittel und die zeiteffiziente Verifikation mit rein formalen Methoden hier, infolge des Komplexitätsanstiegs, nicht immer durchführbar ist. Aus diesem Grund werden bei aktuellen Ansätzen formal beschriebene Modelleigenschaften vorwiegend während der Simulation überprüft. Weiterhin betrachtet die Verifikation immer nur bestimmte Eigenschaften des Systems [Pa09].

Die Begriffe des modellbasierten Entwurfs und des modellbasierten Testens stehen im engen Zusammenhang. In der Dissertation von Conrad „Modell-basierter Test Eingebetteter Software im Automobil“ wurde die folgende Definition des Testens gegeben [Co04]:

> *„Entwicklungsbegleitender Testprozess im Rahmen der Modell-basierten Entwicklung, der eine Kombination unterschiedlicher, sich gut ergänzender Testverfahren umfasst und dabei das ausführbare Modell als reichhaltige Informationsquelle für den Test benutzt“*.

Die modellbasierte Entwicklung bietet die Möglichkeit im Rahmen sogenannter Modelltests ein ausführbares Modell des Systems einer analytischen Qualitätssicherung zu unterziehen [NM10]. Hierbei wird in [RBGW10] zwischen drei verschiedenen Ausprägungen des modellbasierten Testens im Testprozess unterschieden:

- ✓ Beim modellbasierten Testen dienen die Modelle als Grundlage zur Visualisierung von Systemeigenschaften und zur Orientierung für das Testdesign.
- ✓ Die Testartefaktgenerierung aus Modellen wird modellgetriebenes Testen genannt. Die Testfälle werden durch einen Generator erzeugt.
- ✓ Beim modellzentrischen Testen stehen die Modelle im Mittelpunkt der Testaktivität. Sie werden sowohl für Eingangs- als auch Ausgangsinformationen verwendet.

Das elementare Ziel des modellbasierten Testens ist die Verringerung des Integrationsrisikos bei der Integration von Software und Hardware [RDK06]. Auf Basis dessen werden zur Absicherung der Abhängigkeiten zwischen unterschiedlichen Komponenten oder anderen Systemelementen nicht vorhandene Elemente durch Modelle beziehungsweise Simulationen ersetzt [RBGW10]. Ein wichtiges Ziel ist die mögliche Senkung der Entwicklungskosten durch die Verwendung von Modellen zur Generierung und Wartung von Testfällen.

Bei den modellbasierten Tests steht die automatische Testfallerzeugung und Testdatenerzeugung im Vordergrund. In „Practical Model-based Testing“ [UL07] werden die Ansätze zur Testerzeugung in vier unterschiedliche Gruppen geteilt:

1) Erzeugung der Eingangsdaten aus dem Domänenmodell
2) Testfallerzeugung aus dem Umgebungsmodell
3) Erstellung von Testfällen mit Vorhersagen aus dem Verhaltensmodell
4) Erzeugung von Testskripts aus abstrakten Tests

Die unterschiedlichen Testmethoden können bei der Entwicklung auf verschiedene Art und Weise verwendet werden (s. Abschnitt 2.4.4). Der modellbasierte Entwurf sollte durch eine geeignete Kombination verschiedener Testverfahren begleitet werden. Eine zentrale Rolle spielt die Auswahl von Testszenarien unter bestimmten Aktivitäten, weil diese entscheidend für Testumfang und –qualität sind [W12]. In der Arbeit von Conrad [Co04] wurden unterschiedliche bekannte Beschreibungsmittel für Testszenarien dargestellt und eine geeignete Methode für den modellbasierten Test eingebetteter Software gewählt und erweitert. Diese könnte darüber hinaus in anderen Applikationen eingesetzt werden.

2.4.2.3 Modellierungswerkzeuge

Bei der Entwicklung von heutigen Eingebetteten Systemen werden verschiedene Modellierungs- und Spezifikationssprachen, oft in einer gemischten Form eingesetzt.

Eine Modellierungssprache, die die Spezifikation von wichtigen Teilen/Modellen von Hard-Softwaresystemen erlaubt, ist die UML (Unified Modeling Language), obwohl sie für reine, vorzugsweise objektorientierte Softwaresysteme entwickelt wurde [G08]. Die UML lässt sich innerhalb eines Entwurfsprozesses in vielfältiger Weise zur Modellierung von Systemen aus verschiedenen Sichten und zur Modellierung der während des Entwurfes eingesetzten Prozesse nutzen [Zu06]. Durch Notationserweiterungen in Versionen der UML sowie in SysML (Systems Modeling Language) wird die Beschreibung von Signalflüssen, funktionalen Abhängigkeiten und Prozessketten wesentlich verbessert. UML und SysML sind gut als Medium zur Kommunikation zwischen verschiedenen Entwicklergruppen geeignet. Im Buch [K08] wurden unterschiedliche Sichtweisen der Verwendung von UML beziehungsweise SysML dargestellt, z.B. für die Anforderungsanalyse und den Anforderungstest. Die UML hat einen Nachteil: die Semantik der Modelle ist meistens mit informaler, natürlicher Sprache beschrieben. Die dadurch entstehenden Ungenauigkeiten und Mehrdeutigkeiten können zur falschen Modellinterpretation auf der Realisierungsetappe und entsprechend zum inadäquaten Verhalten auf den nachfolgenden Etappen des Lebenszyklus führen [FPM06]. Es existieren Arbeiten zu einer formalen Seman-

tik (z.B. für die UML-Aktivitäts-, Zustands- und Sequenzdiagramme). In [FPM06] werden z.B. Mengen von Zuständen, Transitionen und Flussrelationen, analog zu Definitionen von Petri-Netzen, definiert. Außerdem unterstützt die Benutzung der UML das Datenflussparadigma nur begrenzt. Die UML wird in dieser Arbeit nicht weiter betrachtet.

Im aktuellen Stand der Technik werden in der Mess- und Automatisierungstechnik Entwicklungsumgebungen eingesetzt, die nicht nur einen modellbasierten sondern auch plattformbasierten (und damit plattformspezifischen) Entwurf ermöglichen [Mü12]. Hier kann man graphische Entwurfs- und Simulationswerkzeuge wie MATLAB/Simulink® von Mathworks [M15] oder LabVIEW® von National Instruments [LV15] mit einer steigenden Zahl von Erweiterungen nennen.

Die Entwicklungsumgebung MATLAB® wird sehr häufig mit den eingeschlossenen Teilwerkzeugen Simulink® und Stateflow® verwendet. Diese enthalten verschiedene Möglichkeiten zur Modellierung von komplexen Informationsverarbeitungssystemen [Mü12] [G08]. Simulink® bietet eine graphische Benutzer-Schnittstelle für den Aufbau der Modelle als Blockdiagramme (Datenflussdiagramme). Für die Blöcke sind mathematische Beziehungen von Eingabe, Systemzustand und Ausgabe in Abhängigkeit von der Zeit definiert. Die Verbindungen zwischen den Blöcken beschreiben den Datenfluss zwischen diesen. Komplizierte Modelle profitieren oft durch die Anwendung einer Hierarchie von Subsystemen. Ein Subsystem ist eine Gruppe von Blöcken mit intern verbundenen Schnittstellen sowie Schnittstellen nach außen. Durch Subsysteme ist ein Modell leichter zu lesen und zu verstehen. Stateflow® ist ein graphisches Entwurfs- und Entwicklungswerkzeug für komplizierte Steuerungen und logische Probleme. Mit Stateflow® kann man Stateflow-Diagramme erzeugen, welche prinzipiell eine Variante von endlichen Automaten darstellen, wobei Zustände und Transitionen als Verfeinerung der elementaren Blöcke des Systems verwendet werden. Zusätzlich gibt es speziell integrierte Tools von FPGA-Herstellern, wie z.B. SystemGenerator von Xilinx [Xil11] und DSP-Builder von Altera [Alt15], die speziell für FPGA-Implementierungen Funktionen und Bibliotheken enthalten und die direkte Anbindung zur HDL-Code-Ebene ermöglichen.

Die Umgebung LabVIEW® bietet nicht nur eine native Modellierungs- und Simulationsmöglichkeit, sondern enthält auch Erweiterungen für die plattformspezifische Entwicklung [JH12]. LabVIEW Real-Time (RT) ist eine Kombination aus Entwicklungsumgebung und Zielgerät, die Programme auf einem Universalcontroller mit einem Echtzeit-Betriebssystem abarbeitet. Im Jahr 2003 wurde diese Fähigkeit auf Multi-Core-Systeme und auf modulare Systeme mit FPGA-Modulen erweitert. LabVIEW FPGA ermöglicht eine Realisierung von parallelem Datenfluss auf einem FPGA-Chip, der für den simultanen Ablauf der Operationen (Funktionsblöcke) erzeugt wurde [LV15]. Es gibt die Möglichkeit, mit dem Tool Xilinx System Generator spezielle anwendungsspezifische Funktionen und Bibliotheken zu entwickeln [Xil14][GDMF12]. Die neuen Versionen von

LabVIEW enthalten außerdem ein Statechart Modul für die zustandsbasierte Entwicklung in komplexen Systemen [NI15].

In dieser Arbeit wird die Verwendung von MATLAB/Simulink/Stateflow und LabVIEW bei der prototypischen Entwicklung eines Informationsverarbeitungssystems verfolgt. Hierbei wird aus der grafischen Repräsentierung ein Modell-Code (VHDL oder C-Code) erzeugt, wobei es zumeist auch möglich ist, in Modellelementen Funktionen auf Codeniveau zu entwerfen. Die hardwarespezifischen Werkzeuge übernehmen die anschließende Implementierung. Diese Methodik wird als Transformation von einem plattformunabhängigen zu einem plattformspezifischen Modell verstanden [Mü12]. Der eigentliche Implementierungsvorgang wird vollständig automatisiert.

An dieser Stelle ist es wichtig die oben genannte Hardware-Beschreibungssprache VHDL zu erläutern, weil diese in der Arbeit bei praktischen Realisierungen benutzt wird. In [G08] steht:

> *„Very high-speed integrated circuit description language- VHSIC HDL or VHDL – began life in 1980 under a United States Department of Defense (DoD) requirement for the design of digital circuits following a common design methodology, providing the ability for self-documentation and re-use with new technologies“.*

Die VHDL-Sprache bietet eine formale Semantik, eine sequentielle Programmiersprache und ein Konzept für die FPGA- und/oder ASIC-Implementierung [Ma11]. Die Hauptunterschiede liegen daran, dass erstens der Aufbau einer physikalischen Schaltung modellhaft beschrieben wird, und zweitens die so entstandenen Teile parallel ausgeführt werden. Es existieren frei verfügbare Werkzeuge für Simulation, Optimierung und Synthese. Darüber hinaus besteht die theoretische Möglichkeit der formalen Verifikation durch Anwendung von Techniken aus dem Bereich des symbolischen Model Checking [Ro99][G08].

Als ein Fazit des Abschnittes 2.4.2 muss man anmerken, dass ein durchgängiger modellbasierter Entwurf immer noch eine Herausforderung für die Praxis ist. Außerdem kommen mehrere Modellierungs- und Entwicklungswerkzeuge parallel zum Einsatz, da es bislang keinen geschlossenen professionellen Modellierungsansatz gibt, der alle Entwicklungsphasen von der Anforderungsspezifikation bis zur Implementierung für unterschiedliche Teildisziplinen effizient umsetzt [BST10].

Bei der Beschreibung der konkreten Realisierung eines Informationsverarbeitungssystems in der Wägetechnik wird die Verwendung von LabVIEW mit Erweiterungen detaillierter dargestellt. Es sei angemerkt, dass an diesem Punkt die Notwendigkeit von anderen Entwurfs- und Modellierungswerkzeugen festgestellt wurde (s. Kapitel 5).

2.4.3 Implementierungsarten für Hardware-Software-Systeme

Im Top-Down-Entwicklungsprozess ist die Implementierung die letzte Entwurfsphase. Bevor die tatsächliche Implementierung durchgeführt wird, findet die Modell-zu-Code-

Transformation (auch Codegenerierung genannt [GM07][Be05]) statt. Wie oben geschrieben wurde, wird das plattformspezifische Modell in einen implementierbaren Code umgewandelt (z.B. in C, C++ -Code für DSPs, Mikrocontroller, Softcore-Prozessoren, oder Verilog, VHDL für FPGA-Implementierungen und IP-Cores [G08][Zu06]). Dieser Code ist für den Entwickler im Allgemeinen nicht mehr interpretierbar. Die danach folgende Implementierung für die Zielplattform wird durch herstellerspezifische Werkzeuge, z.B. Compiler und Synthese-Tools, übernommen (bei eigenentwickelten Softcores gilt das für die Programme im Allgemeinen nicht). Die automatische Codegenerierung reduziert Implementierungsfehler, die durch subjektive Einflüsse bei der Umsetzung einer Spezifikation in Programmcode entstehen. Außerdem ist es möglich, eine schnelle Erstellung von Prototypen und die Erhöhung der Wirtschaftlichkeit und Produktivität zu erzielen. Ein großer Nachteil im Zusammenhang mit dieser Arbeit ist die Notwendigkeit von zertifizierten Codegeneratoren und Synthese-Tools, die bei der Entwicklung von sicherheitskritischen und eichfähigen Systemen verwendet werden müssen, da nur diese (zumeist wenigen) von den verschiedenen Zertifizierungsbehörden akzeptiert werden.

Im Weiteren werden die Möglichkeiten der anschließenden Implementierung beschrieben. Die Entwicklung von Eingebetteten Systemen erfolgt effektiv mittels gemeinsamem Entwurf von Hardware- und Software-Komponenten (in Sinne von Hardware-Software-Codesign, s. Abschnitt 2.4.2.2). Nach der Spezifikation muss die Funktionalität eines entwickelten Systems in Software und in Hardware implementiert werden. Je nach Anwendung, Anforderungen und Randbedingungen werden in der Messtechnik häufig Softwarelösungen für ein Eingebettetes Rechnersystem oder reine Hardwarelösungen – oder hybride Systeme aus Soft- und Hardware verwendet [MB07][JH12][Mü12].

2.4.3.1 Konventionelle Arten

Softwarelösungen werden vor allem auf Spezialprozessoren (Mikrocontrollern, Digitalen Signalprozessoren – DSPs, Application-Specific Instruction Set Processors – ASIPs [Tei97][GM07]) oder seltener auf Universalprozessoren implementiert. Heutzutage können Prozessoren zusammen mit Speicherblöcken, Interfaces, weiterer digitaler und eventuell analoger Elektronik auf einem Chip integriert werden. Man spricht in diesem Fall von einem System-on-Chip (SoC) [G08].

In vielen messtechnischen Anwendungsbereichen werden sehr hohe Zeitanforderungen gestellt, die mittels reiner Softwarelösungen nicht zu bewältigen sind. Für diese Fälle sollte nicht nur die Software, sondern auch die Hardware programmierbar sein, um Effektivität und Flexibilität bei der Entwicklung zu erreichen. Dieser Forderung kommen Hardwarekomponenten von modernen Eingebetteten Systemen nach, die als rekonfigurierbare Hardware oder Application-Specific Integrated Circuits (ASICs) realisiert werden [MB07]. ASICs finden den Einsatz in Anwendungen, die in höherer Stückzahl mit möglichst niedrigen Stückkosten produziert werden sollen [Ma11]. Die Forderung nach

hoher Stückzahl resultiert aus den notwendigen Entwicklungskosten. Die ASIC-Lösungen können höhere Performance und höhere Rechenleistungen als die im Weiteren beschriebenen FPGA-Lösungen haben. Diese bilden eine flexible Alternative zu den ASICs und werden seit einigen Jahren verstärkt für anwendungsspezifische Schaltungen eingesetzt [Zu06]. Gegenüber den ASICs sind FPGAs auf Grund der geringeren Kosten der Applikation bei niedrigen Stückzahlen sinnvoll, obwohl der reine Hardwarepreis im Vergleich zu ASICs höher liegt. Der Gesamtsystemaufbau für die Informationsverarbeitung kann in diesen beiden Fällen als System-on-Chip oder als Board-Level-System erfolgen [Mü12].

In dem folgenden Abschnitt werden die Grundlagen der FPGA-Technologie und die Motivation bei der Verwendung von rekonfigurierbarer Hardware dargestellt. Hierbei wird die wichtige Rolle FPGA-basierter Eingebetteter Systeme in der Mess- und Automatisierungstechnik zusammengefasst.

2.4.3.2 FPGA-basierte Systeme

Der Name „*Field Programmable Gate Array-FPGA*“ dieser Architektur ergibt sich aufgrund ihres Aufbau als Array von Gattern und der Programmierbarkeit beim Entwickler beziehungsweise Anwender. Jedes FPGA kann für unterschiedliche Funktionen verwendet werden, wobei der Entwickler den FPGA-Baustein nach individuellen Bedürfnissen anpassen kann [MB07][Zu06].

Die Abbildung 2.8 stellt den Aufbau eines FPGA-Bausteins dar.

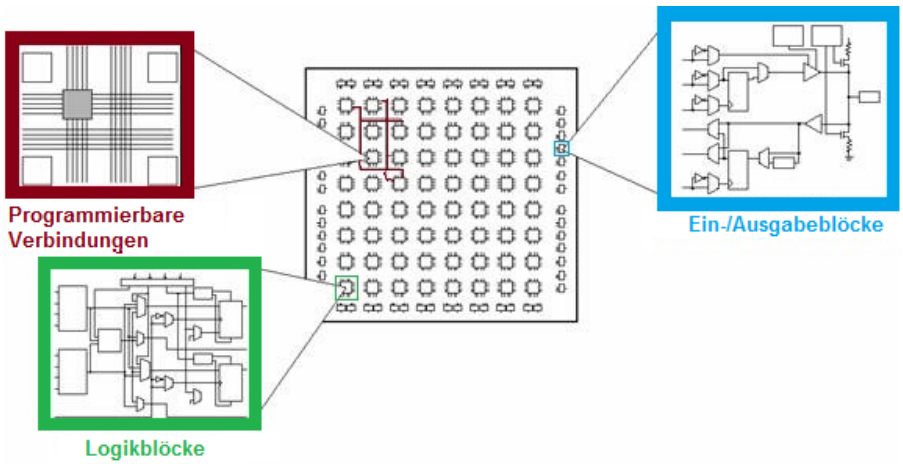


Abbildung 2.8: Bestandteile eines FPGAs, aus [NI14]

Ein FPGA besteht aus vielen Logikblöcken (engl. *logic blocks*), die in einer Matrix angeordnet sind. Diese Logikelemente enthalten die eigentliche Logikfunktion aus Look-up Table (LUT), Flip-Flop (FF) und Multiplexer. Zwischen den Logikblöcken liegen Routing-Kanäle zur Verdrahtung (programmierbare Verbindungen). Das zeitliche

Verhalten ist abhängig von der Platzierung und Verdrahtung (engl. *Place&Route*) der Logikelemente und somit schlecht abschätzbar [GM07].

Abhängig von der Anwendung und den Anforderungen, die eine Applikation verlangt, können unterschiedliche FPGA-Typen eingesetzt werden. Es existiert eine Vielzahl von Typen, die die geeigneten Ressourcen für den konkreten Entwurf bieten. Außer dem inneren Aufbau des Chips können FPGAs in dynamisch beziehungsweise nicht dynamisch, partiell beziehungsweise nicht partiell rekonfigurierbar, aber auch basierend auf ROM, Flash oder RAM unterschieden werden. Bei der Auswahl eines geeigneten FPGA werden Programmiertechnologie, Granularität (Anzahl und Größe der Logikblöcke), Aufbau der Logikblöcke, Art und Anzahl von Verbindungsleitungen sowie Anordnung von Zellen und Verbindungsleitungen berücksichtigt [NM10].

Die modernen FPGAs verfügen inzwischen über eine Vielzahl von Hardware-Ressourcen, die es ermöglichen, mehrere universelle oder auf spezielle Aufgaben angepasste Prozessoren zu realisieren. So werden in komplexen Informationsverarbeitungssystemen zurzeit häufig IP-Cores (Intellectual Property) als vorgefertigte Hardware-Module in die Schaltung integriert [G08]. Als Beispiele von modernen IP-Cores sind Bus-Schnittstellen (z.B. PCI, USB), Mikroprozessoren oder digitale Signalprozessoren (z.B. für Signalverarbeitungsaufgaben wie FFT, Filterung), Kommunikation und Multimedia (z.B. Ethernet, Bluetooth, Mp3) zu nennen. Abhängig von der Anwendung unterscheidet man zwischen Soft-IPs (Softcores) und Hard-IPs (Hardcores). Diese werden von unterschiedlichen Herstellern angeboten und können als anpassbare, parametrisierbare Module oder sogar nur als Black-Box integriert werden [MB07].

Programmentwürfe für FPGAs unterscheiden sich von der Programmierung von Mikrocontrollern oder DSPs. Die Abbildung 2.9 stellt einen Implementierungsprozess für FPGAs dar [GM07].

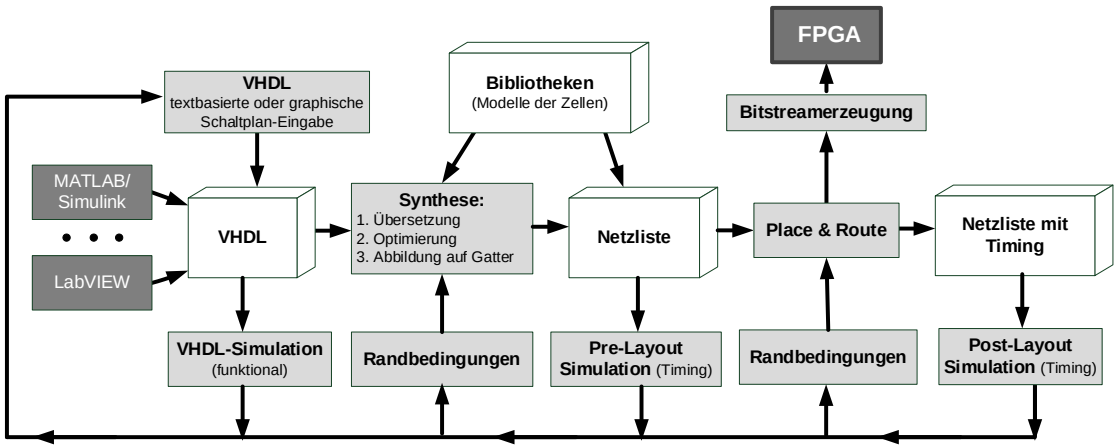


Abbildung 2.9: Ablaufprozess der FPGA-Implementierung

Durch die Beschreibung von Strukturvorschriften mit Hilfe von Hardwarebeschreibungssprachen (z.B. VHDL) wird zunächst die grundlegende Funktionsweise einzelner Blöcke im FPGA und deren Verschaltung untereinander festgelegt. Daraus wird eine Netlist generiert, welche in Textform die Verdrahtung der Schaltung darstellt. Im folgenden Schritt findet die Place&Route Phase statt, die die einzelnen Logikfunktionen auf die FPGA-Ressourcen abbildet. Dies geschieht meistens automatisch mittels Synthesetools, welche die Anordnung der einzelnen Schaltungselemente im Baustein und deren Verdrahtung vorgeben. Der Entwickler kann natürlich durch die Vorgabe verschiedener Bedingungen die Synthese beeinflussen. Weiterhin bietet die Synthese noch die Möglichkeiten zur Simulation. In der Endphase wird ein sogenannter Bitstream als Datei generiert und das FPGA wird durch diese Daten konfiguriert. Dabei wird diese Datei in den internen Konfigurationsspeicher des Bausteins geladen [MB07][Zu06][Ma11].

Die Eingebetteten Systeme auf FPGA-Basis bieten mehrere Vorteile, insbesondere bei der Entwicklung von Prototypen. Die Rekonfigurierbarkeit [NM10] und die Langzeitverfügbarkeit spielen eine wesentliche Rolle. Viele unterschiedliche Ideen und Konzepte können schnell in der Hardware getestet werden, ohne dass zusätzliche Kosten bei der Entwicklung entstehen. FPGAs finden einen breiten Einsatz in Projekten [Mü12][Ma11], da sie ohne großen Zeit- und Kostenaufwand an neue Aufgaben angepasst werden können.

FPGAs werden immer mehr in zeitkritischen Systemen mit harten Echtzeitanforderungen eingesetzt [MB07][JH12]. Dank der gebotenen Flexibilität können selbst komplexe Berechnungen in extrem kurzer Zeit durchgeführt werden. Durch die Möglichkeit in diesen Hardwarestrukturen mit hoher Parallelität zu arbeiten, kann eine massiv parallele Datenverarbeitung realisiert werden. Zusätzlich können FPGAs exakt auf die vorgesehenen Aufgaben zugeschnitten werden und ermöglichen dadurch nicht nur schnelle sondern auch effiziente Systeme [ZKGA13].

In komplexen Informationsverarbeitungssystemen der industriellen und medizinischen Mess- und Automatisierungstechnik, aber auch in der Forschung werden häufig modulare eigebettete Messdatenverarbeitungsplattformen mit rekonfigurierbaren Ein-/Ausgangsmodulen, Eingebetteten Controllern sowie FPGAs eingesetzt. Als Beispiel sollen hier Hersteller wie National Instruments® oder dSPACE® genannt werden [Am10][Mü12][JH12]. Deren Plattformen ermöglichen die Erfassung und die Verarbeitung großer Mengen von Messdaten und stellen eine Vielzahl von Messstellen und Kanälen zur Verfügung. Die direkte Kopplung von FPGA-Modulen mit AD- und DA-Schnittstellen an die Sensoren/Aktoren eines Messsystems ermöglicht eine effektive Realisierung der prozessnahen Signalverarbeitung und der Regelungsfunktionalität in Hardware. Die Prototypen können in kürzerer Zeit mit weniger Aufwand entwickelt werden.

In den späteren Untersuchungen hängt die konkrete Gestaltung der Zielplattform sowohl von den messtechnischen und wirtschaftlichen Anforderungen der Informationsverarbeitung, aber auch von der aktuellen Etappe im Entwicklungsprozess (z.B. Entwicklung eines Prototyps oder der Produktionslösung) ab.

2.4.4 Verifikation und Validierung einschließlich Testen

In den vorangegangenen Abschnitten wurden Definitionen der Begriffe „Verifikation“ und „Validierung“ angegeben. Die Rolle von beiden wurde bei dem modellbasierten Entwurf besonders beschrieben. In diesem Abschnitt werden unterschiedliche Verifikationsmethoden und Testmöglichkeiten detaillierter präsentiert.

Bei Entwicklung einer komplexen Informationsverarbeitungslösung, insbesondere in sicherheitskritischen und eichfähigen messtechnischen Systemen, benötigen die Verifikation und Validierung oft einen besonderen Aufwand und stellen eine Herausforderung dar. Es wird ein Prozess benötigt, der die auf den domänenspezifischen Entwicklungsprozess abgestimmten Verifikations- und Validierungsaktivitäten mit den verwendeten Methoden, Techniken und Werkzeugen enthält. Dazu werden die Wechselwirkungen definiert, die zwischen den Entwicklungs- und Verifikationsprozessen auf Grund von gemeinsam genutzter Information entstehen.

Es existieren zahlreiche Verifikations- und Validierungsansätze, die in der Literatur meistens einzeln zu finden sind. Die Abbildung 2.10 stellt die Hauptmethoden der Verifikation und Validierung im behandelten Entwicklungskontext in Gruppen dar (angelehnt an [Ma11][CK13][Lü09][NM10]).

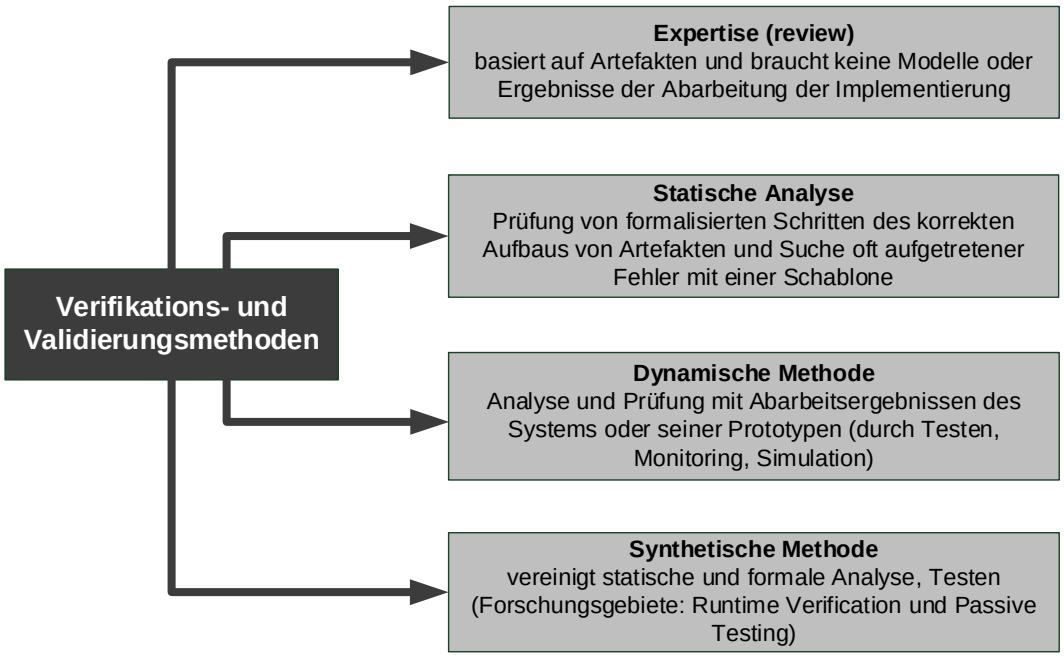


Abbildung 2.10: Klassifizierung von Verifikations- und Validierungsmethoden

In Eingebetteten Systemen werden häufig die Simulation und die Methoden der formalen Verifikation verwendet.

Im Grunde genommen ist eine Simulation ein experimentelles Durchführen eines Testszenarios am Modell in unterschiedlichen Entwurfsphasen und mit verschiedenen Detailangaben [W12][MA11]. Für Eingebettete Systeme kommen verschiedene Simulationsarten zum Einsatz: Simulation auf höherer Modellebene (z.B. MATLAB/Simulink, LabVIEW), HDL-Simulation, „Pre-Layout"- und „Post-Layout"-Simulation [MB07][Zu06].

Ein Verfahren, das zur Verifikation einer Funktion anhand von mathematischen (formalen) Methoden durchgeführt wird, wird als formale Verifikation bezeichnet [Tei97][Kl09]. Eine Voraussetzung der erfolgreichen Anwendung dieser Methode ist eine formale Beschreibung des Testobjektes (z.B. Schaltungsdarstellung in einer Hardware-Beschreibungssprache, Strukturdarstellung in Form von Netzlisten). Formale Methoden bieten Möglichkeiten der mathematischen Analyse und können dabei die genauere Untersuchung von bestimmten Systemeigenschaften garantieren (z.B. Model Checking, Equivalence Checking). [NM10][Ma11]

Weiterhin existiert die Methode der Assertion-Based Verifikation. Dabei werden die Eigenschaften dergestalt definiert, dass sowohl die Simulation als auch die formale Verifikation verwendet werden können [VA14][LPJW08].

Die Arbeit von Pacholik [Pa09] stellt unterschiedliche Methoden, Techniken und Werkzeuge zu einem Verifikationsprozess zusammen. Diese Dissertation beschäftigt sich mit der Entwicklung und Gegenüberstellung von Methoden zur automatisierten Verifikation von ausführbaren Systemspezifikationen. In diesem Zusammenhang wurde vom Autor festgestellt, dass sich in Bezug auf den Verifikationsprozess bisher keine geeignete formale Spezifikationssprache vollständig in der Praxis durchsetzen konnte. Die Ansätze wurden vor allem für die Verifikation temporaler Eigenschaften in komplexen ausführbaren Spezifikationen entwickelt.

Um die Risiken der Entwicklung und den großen Aufwand bei der funktionalen Verifikation von komplexen Informationsverarbeitungssystemen zu minimieren, sollte bei der Hardwareimplementierung weitgehend auf verifizierte Teilkomponenten zurückgegriffen werden (z.B. verifizierter Softcore [Sch12]). Dafür ist es sinnvoll, dass eine eigenschaftserhaltende Komposition existiert [He08].

Im Zusammenhang mit den Begriffen der Verifikation und der Validierung wird häufig der Begriff des Tests verwendet. Beim Testen werden das System und seine Teile überprüft, um mögliche Fehlwirkungen oder Fehlverhalten festzustellen. Das Hauptziel ist eine Analyse zur Ermittlung des Fehlerverhaltens des fertigen Systems, um dabei ein inkorrektes Verhalten möglichst zu vermeiden. Des Weiteren dienen Tests auch dazu, mehr Informationen über das zu entwickelnde System zu erlangen.

Das Testen stellt eine hohe Herausforderung bei der Entwicklung von modernen Eingebetteten Systemen dar. Es gibt viele unterschiedliche Klassifikationen von Testmethoden. Diese lassen sich abhängig von den betrachteten Sachverhalten gliedern [SRWL11].

Bei der Betrachtung von aktuellen ***Testzielen*** der Entwicklungs- beziehungsweise Betriebsphase lassen sich die Testmethoden in strukturelle, funktionale und nichtfunktionale Tests unterteilen [Neu04]. Das ***statische Testen*** (engl. „*static testing*“) ist ein Verfahren, das ohne eine Ausführung des zu testenden Systems auskommt. Damit lassen sich in frühen Entwicklungsphasen mögliche Fehler lokalisieren. Das statische Testen wird häufig auch als Nachprüfung (engl. „*review*“) bezeichnet. ***Strukturelle Tests*** **(**engl. „*structural tests*“) werden zu den dynamischen Testverfahren gezählt und haben dabei das Ziel die Struktur, z.B. den Kontroll- oder Datenfluss, zu untersuchen. Aus diesem Grund muss die interne Funktions- und Verhaltensweise bekannt sein. Deshalb ist dieses Verfahren auch in die Gruppe „*white-box*“ oder auch „*glass-box*“ Tests eingeordnet. Das Testen geschieht durch gesteuertes Ausführen der zu testenden Objekte bei geeigneter Stimulation der Eingänge und der Auswertung von Zwischen- und Endergebnissen und der Analyse des dabei auftretenden Steuerflusses. ***Funktionale Testverfahren*** (engl. „*functional tests*“) basieren auf den vorgegebenen Spezifikationen und gehören ebenfalls zur Gruppe der dynamischen Testmethoden. Dazu sind keinerlei Informationen über die interne Struktur relevant. Daher werden diese Verfahren auch als „*black-box*“ Tests bezeichnet. Zu den funktionalen Tests zählen die Überprüfungen von Sicherheitsfunktionen. Beim ***nichtfunktionalen Testen*** (engl. „*non-functional tests*“) wird das System gegen nichtfunktionale Anforderungen überprüft, z.B. auf Zuverlässigkeit oder Leistungsparameter. In der Regel handelt es sich bei diesen Verfahren um „*black-box*“ Tests. Im Einzelfall kann es allerdings notwendig sein, bestimmte interne Parameter wie Taktraten oder systeminterne Zeitparameter zu kennen. Daher werden die nichtfunktionalen Tests auch als „*grey-box*“ Tests bezeichnet. [GT02][SRWL11][RDK06]

Das zu testende System wird aus der Sicht des ***Testbereiches*** unterteilt. In der Regel besteht ein System aus einer größeren Anzahl von verschiedenen Komponenten, die im Zusammenspiel für das Verhalten des Gesamtsystems verantwortlich sind. Diese können getrennt oder in Kombination miteinander getestet werden. Auch die Umgebungskomponenten, z.B. mechanische oder elektrische Module, können mit speziellen Tests untersucht werden [Bie07]. Aus der Sicht eines Systemteils ist es möglich, das Gesamtsystem zu überprüfen. Das damit verbundene Testen verschiedener Bereiche ermöglicht die Entdeckung unterschiedlicher Fehler. Es wird zwischen Komponententest, Integrationstest, Systemtest und Abnahmetest unterschieden [SRWL11]. Beim ***Komponententest*** wird eine kleinste testbare Einheit des Gesamtsystems, z.B. ein Mikrocontroller, isoliert getestet. Beim ***Integrationstest*** werden sehr eng zusammengehörige Komponenten getestet. Hierbei ist es möglich, Fehler in Verbindungen und in der Zusammenarbeit zu entdecken. Das ganze System wird mit einem ***Systemtestverfahren*** überprüft. Allerdings werden

häufig komplexe Eingebettete Systeme im Vorfeld in Subsysteme unterteilt. Hierzu wird das System über verschiedene Schnittstellen auf Funktionalität, Performance und Skalierbarkeit überprüft. Der Abnahmetest (auch als Akzeptanztest bezeichnet) ist Bestandteil der Abnahme durch den Auftraggeber, der unter Umständen mit eigenen Anforderungslisten den Test gegenprüft. [UL07]

Die Testmethoden unterscheiden sich abhängig von den verwendeten ***Testdurchführungsplattformen***. Die Testplattform, die entsprechend dem Testziel und der Testumgebung gewählt wird, erzeugt Eingangsgrößen für das Testobjekt. Anschließend werden die Ergebnisse beobachtet und analysiert. Das bekannte Verfahren ***Model in the Loop (MiL)*** wird vorwiegend in den oberen Abstraktionsebenen des Entwurfs genutzt. Dabei ist das Testen in früheren Entwicklungsphasen möglich. Man spricht von MiL, wenn das System mit einem Umgebungsmodell simuliert und getestet wird. Dazu wird keine Hardware verwendet. Ein weiteres Verfahren, bei dem die Software des zu testenden Systems kontrolliert wird, ist als ***Software in the Loop (SiL)*** bekannt. Dabei wird die Software nicht auf der Zielhardware ausgeführt, sondern auf einem Simulator, z.B. auf einem Desktopcomputer. SiL benötigt einen generierten oder geschriebenen Code (verfeinerter Entwurf) beziehungsweise ein ausführbares Modell. Deshalb wird dieses Verfahren vorwiegend auf niedrigeren Abstraktionsebenen benutzt. Das ***Processor in the Loop (PiL)*** – Verfahren ermöglicht, Fehler basierend auf dem Zielhardwarecompiler oder der Zielprozessorarchitektur zu finden. In diesem Fall wird die Software auf der Zielhardware und dem Zielprozessor oder einem Zielprozessoremulator ausgeführt. Die Einflüsse der Umgebung sind weiterhin in Modellform vorhanden und werden simuliert. Ein weiteres Verfahren, bei dem die Software auf der finalen Zielhardware läuft und die Umgebung mithilfe eines echtzeitfähigen Modells simuliert wird, ist ***Hardware in the Loop (HiL)***. HiL verlangt ein vorhandenes Zielsystem, die implementierte Software und die reale Umgebung. Dieses wird bei allen Tests benutzt, die das implementierte Zielsystem (beginnend mit dem Modultest bis zum Abnahmetest) behandeln. Durch HiL können die Schnittstellen und sogenannte „*low-level*“ Dienste des Prozessors (z.B. Lesen oder Schreiben des Speichers) überprüft werden. Beim letzten Integrationsniveau wird das entwickelte Eingebettete System in der realen Zielumgebung (mit dem Einbettenden System) geprüft. Das Auslösen und Beobachten von Fehlern ist häufig aufwendig, da der Zugriff auf interne Signale schwer bis unmöglich ist. [Co04][RBGW10]

Außer den oben genannten Methoden existieren die reaktiven Tests, die sich in zwei grundlegend verwendete Konfigurationen aufteilen. Die ***„open-loop“ Konfiguration*** wird gewählt, wenn Eingangs- und Ausgangsverhalten überprüft werden sollen. Eine oder mehrere Eingangsgrößen beeinflussen durch das Systemverhalten die Ausgangsgrößen. Die Eingangsgrößen können vom Prozess oder von einem Testdatengenerator kommen. Die zweite ***Konfiguration „closed-loop“*** wird vorwiegend beim Regeln und Steuern von

Prozessen verwendet [Lü09]. Im ersten Fall wird die Regelgröße erfasst und mit der Führungsgröße verglichen, mit dem Ziel die Regelgröße an die Führungsgröße anzugleichen. In diesem Testverfahren wird für die Umgebung ein Modell oder der reale Prozess verwendet. Wenn möglich werden dazu reale Sensoren und Aktoren benutzt [BK08]. Beim Steuern werden binäre Ausgangssignale in Abhängigkeit binärer Eingangssignale und Zustandsvariablen des Steueralgorithmus, der z.B. mit Statecharts beschrieben wird, erzeugt. Regelung und Steuerung können in komplexeren Systemen gemischt auftreten.

Die Testverfahren sind mit den Testaktivitäten im Entwicklungsprozess verbunden [Mü12]. Es können verschiedene Verfahren für eine Entwicklungsetappe geeignet oder weniger bis gar nicht anwendbar sein. Besonders für komplexe Eingebettete Systeme gehen die Einteilungen teilweise fließend ineinander über und können sich überlagern. Es muss ein geeignetes Konzept abhängig von der Applikation gefunden werden.

2.4.5 Produktlinien

In Bezug auf die Entwicklung von komplexen Eingebetteten Systemen für die Informationsverarbeitungsaufgaben in der Messtechnik ist die Betrachtung von Produktlinien von Interesse. Eine Produktlinie setzt sich aus einer Reihe verschiedener Produkte zusammen, die eine Menge von Eigenschaften miteinander teilen und dadurch auf ein bestimmtes Marktsegment abzielen. Sie gehören damit zusammen zu einer Anwendungsdomäne. Eine Produktlinie besteht demnach aus einer Menge von Produkten, die einen gemeinsamen Kern von Merkmalen haben und die bei einer Reihe von Eigenschaften unterschiedliche Ausprägungen besitzen und somit in diesen Eigenschaften variieren. [LSR07]

In den letzten zehn Jahren haben sich Software-Produktlinien für verschiedene Domänen als ein vielversprechendes Paradigma der Softwareentwicklung verbreitet [PBL05]. Die gemeinsamen Eigenschaften der Software-Produkte werden durch eine gemeinsame Basis (Core beziehungsweise Plattform) realisiert. Eine Plattform ist eine Basis von Technologien, auf denen andere Technologien oder Prozesse aufgebaut sind. Die Integration von Variabilität in dieser gemeinsamen Plattform ermöglicht die effiziente Entwicklung von individuellen angepassten Software-Produkten. Der hohe Anteil der Wiederverwendung führt zu einer Reduktion von Entwicklungszeit und Entwicklungskosten bei gleichzeitig steigender Software-Qualität [BKPS04]. Produktlinienentwurf verlangt eine produktlinienorientierte Anforderungsanalyse. Die Ansätze dazu findet man in [St04].

Produktlinien sind besonders in der Automobilindustrie verbreitet. Hierbei erscheinen als Merkmale (*features*) der Produktlinie Software-, Hardware- und gerätetechnische Komponenten [LSR07].

Beim Produktlinienentwurf werden Applikationen mit verschiedenen Merkmalen aus einem Entwurfsergebnis in einer bestimmten Domäne abgeleitet, die eine Übermenge die-

ser Merkmale enthält. Das Domain Engineering ist ein Prozess im Produktlinien-Engineering, in dem die Gemeinsamkeiten und die Variabilität der Produktlinie definiert und realisiert werden. Der Bereich des Domain Engineering umfasst alle Aktivitäten, die die gesamte Produktlinie betreffen. Hier werden die Anforderungen, die theoretisch alle Produkte der Produktlinie betreffen, festgehalten. Domain Engineering schafft eine Plattform, die die Abstrahierung von einzelnen Produkten ermöglicht. Bei dem Application Engineering werden kundenspezifische Produkte unter Ausnutzung der Produktlinien-Variabilität realisiert. Eine Ausdifferenzierung der Produkte wird durch die entsprechende Bindung der Variationspunkte mit den vordefinierten Varianten oder mit den entwickelten Kundenvarianten ermöglicht. [LSR07]

Um die erforderliche Qualitätssicherung zu erreichen, ist das Testen von Software-Produktlinien von großer Bedeutung. Bei den bekannten Verfahren für automatische Testfallgenerierung werden die Testfälle aus einem Testmodell der Domäne abgeleitet. [Fa10]

Die Verwendung von Produktlinien für wägetechnische Systeme ist dann sinnvoll, wenn Wägezellen des gleichen Grundtyps an verschiedene Maschinen (z.B. Verpackungsmaschinen, Dosierwaagen in der pharmazeutischen Industrie), in die sie eingebaut werden, anzupassen sind. In der Dissertation von Farrag [Fa10] wurde ein Referenzprozess beschrieben, in dem der Entwurf mit Produktlinien verallgemeinert dargestellt wird. Außerdem wird die Verwendung von gefärbten Statecharts (Colored State Charts) zum Testen von Software-Produktlinien vorgeschlagen. Diese Grundansätze können auch für die Entwicklung von Eingebetteten Systemen in der Messtechnik verwendet werden. In den messtechnischen Anwendungen von Produktlinien ist es sinnvoll, beim Domain Engineering einschließlich Variabilität und der Applikationsentwicklung nicht nur die Software sondern die gesamte Informationsverarbeitung und das Einbettende System (Elektromechanik, Optik usw.) mit zu berücksichtigen.

2.5 Zusammenfassung und Ableitung der eigenen Arbeiten

Im Rahmen des Kapitels 2 wurden die allgemeinen Aspekte der Informationsverarbeitung in der Messtechnik betrachtet. In Verallgemeinerung der Erfahrungen wird in dieser Dissertation eine Methode zum modellbasierten Entwurf entwickelt. Der Anwendungsschwerpunkt liegt in der dynamischen Wägetechnik, deswegen wurden die Grundlagen, basierend auf dieser Domäne, detaillierter dargestellt. Anhand dieses Schwerpunktes wird die entwickelte Methode vertieft.

Einführend enthält der Abschnitt 2.1 die Grundbegriffe und die für diese Arbeit wichtigsten Definitionen in messtechnischen Systemen. Hierbei wurde zuerst auf die Aspekte des Messens, die technischen Parameter und die messtechnischen Eigenschaften eingegangen. Als Beispiel dienen die Unterschiede von Kraft- und Massebestimmung. Nach der

Definition eines Messgerätes folgt eine Diskussion der verschiedenen Ursachen für Messabweichungen bei der Abschätzung der Messung. Dabei stellt der Abschnitt Informationen zu Arten von Messfehlern und zur Fehlerberechnung dar.

Die wesentlichen speziellen Anforderungen in der Messtechnik insbesondere in den eichfähigen Systemen, wie diese in der dynamischen Wägetechnik vorliegen, wurden im Abschnitt 2.2 genannt. Hierzu spielen unterschiedliche Gesetzte, Vorschriften und Richtlinien eine wichtige Rolle. *Die in der vorliegenden Dissertation entwickelte Methodik verlangt, diese Anforderungen an das Messgerät schon bei der prototypischen Entwicklung zu berücksichtigen, um einen zertifizierbaren Prozess zur Informationsverarbeitung zu erhalten. Das auf dem Prototyp basierende produzierbare Erzeugnis kann dann deutlich einfacher und effektiver für die o.g. Forderungen entwickelt werden.*

Der Abschnitt 2.3 unterteilt sich in die zwei Unterkapitel. Zuerst wurden die bestehenden Prinzipien der digitalen Informationsverarbeitung in der Mess- und Wägetechnik zusammengefasst. Dabei wurden deren Vorteile kurz erläutert. Für das in dieser Arbeit beispielhaft betrachtete Wägesystem – eine EMK-Waage – wurde eine gründliche Recherche zu in der Signalverarbeitung verwendeten und benötigten Algorithmen durchgeführt. Diese ist ebenfalls im ersten Unterkapitel beschrieben. Das zweite Unterkapitel umfasst die Besonderheiten der digitalen Informationsverarbeitung unter dem Gesichtspunkt der Eichfähigkeit und anderer Empfehlungen/Forderungen. Diese haben eine besondere Relevanz für diese Arbeit. Das ergibt sich zum einen aus deren Bedeutung in der Domäne sowie aus der Berücksichtigung von speziellen Anforderungen an die Informationsverarbeitung, als Teil des gesamten Messsystems. Die im Unterkapitel 2.3.2 genannten Standards enthalten meistens Forderungen zur entwickelten Software und zu deren Dokumentierung. *Für rekonfigurierbare Hardware gibt es zwar Standards zur qualitätssichernden Entwicklung, aber nicht im Zusammenhang mit der Messtechnik, speziell der eichfähigen dynamischen Wägetechnik. Dieses ist deshalb Bearbeitungsgegenstand in der Dissertation.*

Der Schwerpunkt dieser Arbeit liegt in der Entwicklung eines speziellen für die messtechnische Domäne angepassten Prozesses, der die Vorgehensweise beim Entwurf von Eingebetteten Systemen für die Informationsverarbeitung beschreibt. Außerdem soll dieser Prozess nachweisen, dass die speziellen Anforderungen an eichfähige Messtechnik und damit verbundene weitere Anforderungen realisiert werden, und damit zertifizierbar sein. In diesem Zusammenhang wurde eine gründliche Recherche zu im Prozess-Engineering verbreiteten Entwicklungsvorgehensweisen in der Softwaretechnik durchgeführt. Im Unterkapitel 2.4.1 sind deren Vor- und Nachteile gegeben. Im Grunde enthält ein Entwicklungsprozess unterschiedliche Entwicklungsetappen, wie Anforderungsebene, Spezifikationsebene, Entwurfsebene und Implementierungsebene, in denen jeweils verschiedene Ziele verfolgt werden. Um die Qualität des Entwurfes zu verbessern und die Wechselwirkungen der einzelnen Etappen zu analysieren, wird die Modellbasierte Entwicklung

durch ausführbare Modelle als Stand der Technik genutzt. Dabei ist es möglich, Validierung und Verifikation in früheren Entwicklungsphasen durchzuführen. *Als Ansatzpunkt dieser Arbeit wird das Vorgehen auf Basis von W-Modellen nach Herzlich und Spillner für die Entwicklung eines Informationsverarbeitungssystems in der genannten Domäne bevorzugt, wobei eine Reihe von Modifikationen und Erweiterungen realisiert wurden und im Weiteren beschrieben sind.*

Für die Implementierung einer Informationsverarbeitung mit Eingebetteten Systemen sind konventionelle Arten und/oder FPGA-basierte Systeme als Zielplattform anwendbar. Im Unterkapitel 2.4.3.2 wurden Informationen zur FPGA-Technologie und die Vorteile der Verwendung dieser Plattformen unter anderem zur Realisierung von zeitkritischen Aufgaben beschrieben. Eine effektive Realisierung von Prototypen mit prozessnaher Signalverarbeitung in kürzerer Zeit und mit weniger Aufwand ermöglichen modulare eingebettete Messdatenverarbeitungsplattformen. Diese können rekonfigurierbare Ein-/Ausgangsmodule (enthalten E/A-Schnittstellen mit rekonfigurierbarer Logik), eingebettete Controller sowie FPGAs besitzen. Damit wird der Einsatz einer modularen FPGA-basierten Messdatenverarbeitungsplattform bei der Entwicklung eines Prototyps in den vorliegenden Untersuchungen motiviert.

Die Methodik beinhaltet folgende Prämissen und Zielstellungen: Ein Prototyp wird modellbasiert entwickelt und nutzt ein Eingebettetes System auf Basis eines kostenintensiven und leistungsfähigen FPGA. Die konkrete Gestaltung der Zielplattform soll sowohl von den messtechnischen und wirtschaftlichen Anforderungen der Informationsverarbeitung als auch von der aktuellen Etappe im Entwicklungsprozess abhängig sein. Die Aspekte einer produzierbaren Lösung mit entsprechenden Anforderungen müssen im gesamten Entwicklungsprozess schon bei der prototypischen Untersuchung berücksichtigt werden.

Ausgehend von den Vorgehensweisen bei der Entwicklung von komplexen Informationsverarbeitungssystemen wurde die Notwendigkeit von Validierungs-, Verifikationstechniken und Testen in früheren Entwurfsphasen erläutert. *Abhängig von der Etappe der Entwicklung können verschiedene Verfahren indiziert oder kontraindiziert, aber auch teilweise verknüpft werden. Validier- und Verifizierbarkeit für den Entwicklungsprozess sind als Entwurfskriterien im gesamten Prozess zu berücksichtigen. Es muss dabei ein geeignetes Konzept für die Anwendung in der Messtechnik gefunden werden. Geeignete Formen von Wiederverwendung sind als integraler Bestandteil zu berücksichtigen.*

Ein weiterer interessanter Aspekt, der in der Arbeit ansatzweise bearbeitet wird, beschäftigt sich mit dem Produktlinienentwurf. Dabei kann man von einem Entwurf für die messtechnische Domäne Applikationen mit verschiedenen Merkmalen ableiten. Das ist z.B. in der Wägetechnik sinnvoll, wenn Wägezellen des gleichen Grundtyps an verschiedene Anwendungen anzupassen sind.

Das für Software etablierte Konzept ist in der Dissertation teilweise für rekonfigurierbare Hardware zu entwickeln. Auch hier sind effektive Test- und Verifikationsstrategien zu integrieren.

Im Weiteren wird das Konzept eines Prozesses für die Entwicklung von komplexen Informationsverarbeitungssystemen beschrieben, der die eben genannten Zielstellungen umsetzt. Dieser dient in erster Linie der Qualitätssicherung auf organisatorischen und technischen Ebenen. Der Entwicklungsprozess soll erreichen, dass die Aktivitäten in korrekter Reihenfolge mit entsprechenden Anforderungen durchgeführt werden. Außerdem müssen die möglichen Fehler bei der Anforderungsdefinition, in der Spezifikation, den groben und verfeinerten Entwürfen und bei der Implementierung möglichst früh erkannt und beseitigt werden. Das verlangt auch die Schaffung effektiver Vorgehensweisen auf den hierarchisch aufeinander aufbauenden Entwurfsebenen und mit deren Wechselwirkungen.

3 Zertifizierbarer Entwicklungsprozesses für Informationsverarbeitungssysteme in der Messtechnik (ZEfIRA)

In diesem Abschnitt wird der für die vorliegende Arbeit entwickelte „**Z**ertifizierbare **E**ntwicklungsprozess **f**ür **I**nformationsverarbeitungssysteme mit speziellen **R**echner**a**rchitekturen“ (im Weiteren kurz ZEfIRA genannt) präsentiert.

Das Konzept von ZEfIRA ist eine spezielle Vorgehensweise für die Entwicklung von Hard- und Software zur komplexen eingebetteten Informationsverarbeitung in der Messtechnik, hier insbesondere beeinflusst durch die Wägetechnik. Es stellt den groben Rahmen eines zertifizierbaren Prozesses in der benannten Domäne dar, wobei Zertifizierbarkeit bereits im Abschnitt 2.4 eingeführt wurde. Beginnend mit der Analyse eines eventuell vorhandenes Vorläufersystems über eine modellbasierte prototypische Entwicklung zu einer produzierbaren Lösung (Produkt) wird das Konzept ZEfIRA auf Basis eines so genannten 3W-Modells systematisch beschrieben (Abbildung 3.1).

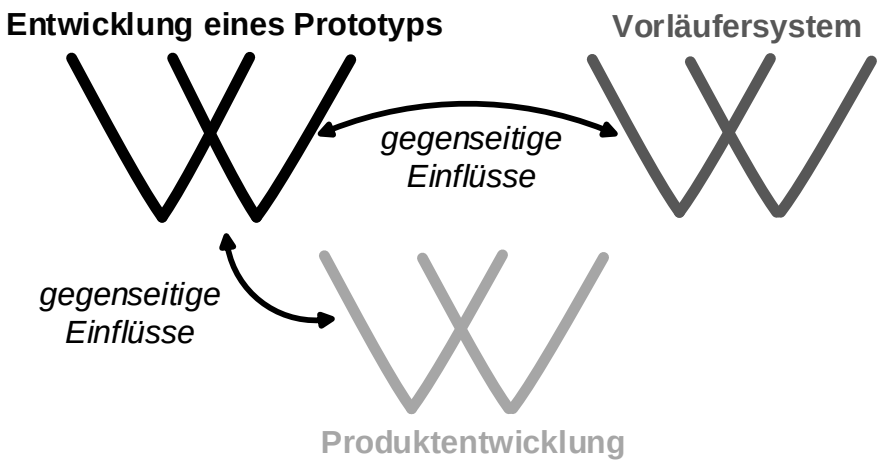


Abbildung 3.1: Konzept des 3W-Modells (allgemein)

Unter dem Vorläufersystem wird eine Informationsverarbeitungslösung eines messtechnischen Produktes verstanden, welches prinzipiell ähnliche Funktionen realisiert, wie diejenigen die das aktuell zu entwickelnde Produkt enthalten soll. Die Prototypenentwicklung hat diese Aufgaben: frühzeitiger Nachweis von spezifischen Produkteigenschaften, genauere Spezifikation von unklaren Anforderungen, Demonstration und Validierung von Entwürfen (s. Abschnitt 2.4). Sie ermöglicht die Akzeptanz bei Kunden/Endbenutzern bereits in den früheren Entwicklungsphasen. Weiterhin soll hier die Entwicklung des Prototyps dazu dienen, verschiedene Lösungsvarianten nicht nur bezüglich dieser Akzeptanz, sondern bezüglich der weitgehend optimalen Realisierung der funktionalen Eigenschaften zu untersuchen. In diesem Zusammenhang bezieht sich „optimal“ sowohl auf

technische als auch wirtschaftliche Parameter des künftigen Produkts. Die Produktentwicklung soll die mit dem Prototyp gefundene Realisierung, unter den endgültigen technischen, technologischen und wirtschaftlichen Parametern, als weitgehend optimale Lösung liefern.

Die Grundideen des Prozesses sind:

- ✓ Teilweise parallele Durchführung der Analyse des Vorgängersystems (falls es vorhanden ist und mit den gleichen oder ähnlichen Modellen entwickelt wurde), der modellbasierten Entwicklung eines Prototyps und der Überführung in ein wirtschaftlich verwertbares Produkt. Es entstehen dadurch die drei W-Abläufe.
- ✓ Frühe Testphasen mit hierarchisch aufeinander aufbauenden Modellen, Ableitung der Testfälle des implementierten Systems auf Basis dieser modellbasierten Tests (linke Seite der W-Vorgehensweisen).
- ✓ Korrektur von gefundenen Fehlen auf der Hierarchieebene, auf der sie entstanden sind, sowohl bei den modell- als auch den zielsystembasierenden Tests.
- ✓ Gleichzeitige Entwicklung von messtechnischem Teil, herkömmlicher und rekonfigurierbarer Hardware, eventuell Softcore-Prozessoren und Universalprozessoren/Digital Signalprozessoren (einschließlich Software). Durch die parallele Entwicklung entstehen regelmäßig veränderliche Anforderungen für die Informationsverarbeitung auf Grund der noch vorhandenen Lösungsvariabilität der einzelnen Teillinien (Eingebettetes System) und des messtechnischen Einbettenden Systems. Weiterhin geht es um eine gemeinsame Entwicklung in den früheren Phasen, unterschiedliche Partitionierungszeitpunkte aller Teillinien und zumeist die Kopplung von mehreren Modellierungssprachen (linke Seite der W-Vorgehensweisen).
- ✓ Der Prozess beabsichtigt, dass Erkenntnisse und Lösungen bei der Entwicklung des neuen Erzeugnisses auch in das alte Erzeugnis übertragen werden, falls dieses parallel in einer weiter entwickelten Form produziert wird (Einflüsse zwischen dem linken und dem rechten W).
- ✓ Integration der Forderungen zur Metrologischen Sicherheit und Eichfähigkeit schon in den modellbasierten Entwurfsprozess, beginnend beim Prototyp und fortgesetzt bei der Realisierung des produzierbaren Erzeugnisses.
- ✓ Betrachtung der Möglichkeiten für die Entwicklung eines produzierbaren Erzeugnisses bereits bei der Prototypenentwicklung (Einflüsse zwischen dem linken und dem unteren W).

Die hier genannten Ideen sind teilweise in W-Entwicklungsprozessen anderer Anwendungsdomänen [HSD09][SRWL11] zu finden. Dennoch ist die Vollständigkeit bei Berücksichtigung von gleichzeitiger Nutzung rekonfigurierbarer Hardware und den speziellen gesetzlichen Anforderungen und Richtlinien in der Messtechnik nicht gegeben.

Durch das systematische Vorgehen im Rahmen des entwickelten Prozessmodells wird davon ausgegangen, dass der Prozess zertifizierbar wird, d.h. mit hoher Wahrscheinlichkeit von einer entsprechenden Einrichtung zertifiziert wird (z.B. nach FDA-Anforderungen, siehe auch Abschnitt 2.3.2).

Im Weiteren werden die Teilprozesse (W-Modelle) und deren Beziehungen im Konzept ZEfIRA veranschaulicht. Die Anforderungen aus der Zieldomäne (Messtechnik, speziell auch Wägetechnik) werden dabei als Basis genommen.

3.1 W-Prozess zur Entwicklung eines Prototyps

Bei der Entwicklung von komplexen Systemen ist es üblich neue Konzepte, Ideen und Zielplattformen zu erproben, bevor eine fertige produzierbare Lösung in einer Großserie ausgefertigt wird. Es wird dabei ein Prototyp beziehungsweise ein Demonstrator als ein einzelnes Exemplar entwickelt, das dem geplanten Produkt (hier Messgerät) vor allem in den kritischen Merkmalen entspricht.

Die Entwicklung von Prototypen ermöglicht es, aus mehreren Realisierungs- und Implementierungsvarianten eine passende, für das konkrete Messsystem möglichst optimale Entwurfslösung (z.B. Einhaltung der aus dem Frequenzspektrum des messtechnischen Teils resultierenden Latenzen bei gleichzeitig minimalem Ressourcenverbrauch) zu finden. Dieses wurde bereits im Abschnitt 2.4 näher erläutert. Das für Eingebettete Systeme geeignete Vorgehensmodell ist ein für die vorliegende Arbeit weiterentwickeltes W-Modell (Abb. 3.2). In diesem wird der modellbasierte Top-Down-Entwurf mit integriertem modellbasiertem Testen realisiert, der in den höhereren Abstraktionsebenen mit nicht partitionierten Modellen beginnt (linke Seite des W‘s). Dieses Modell ist angelehnt an Herzlich´s und Spillner´s W-Modelle, wobei die positiven Aspekte beider kombiniert werden und Erweiterungen für das Zielgebiet erfolgen. Die Erweiterungen sind nötig, da der modellbasierte Entwurf Eingebetteter Systeme, auch unter Verwendung rekonfigurierbarer Hardware komplexere Anforderungen bei der Systementwicklung stellt.

Die funktionalen Anforderungen für ein zu entwerfendes System werden in ein Modell überführt (eventuell in ein ausführbares Spezifikationsmodell beziehungsweise in einen virtuellen Prototyp beim technischen Systementwurf [SFM12][Mü12]). Die Etappen bis zur Modulspezifikation (nach der Partitionierung) sind zielsystemunabhängig modelliert, untersucht und getestet. Das „Testen“ soll hier als Erweiterung um zusätzliche Validierungs- und Verifikationsaktivitäten verstanden werden. Die zu verwendenden Modelle müssen in einem geeigneten Werkzeug ausführbar sein (auf Modellniveau testbar). Ein gutes Beispiel ist die Verwendung von MATLAB/Simulink [Mü12]. Für die Verifikation ist es sinnvoll, aus den simulierbaren Beschreibungen Modelle zu generieren, die dieser zugänglich sind (z.B. Petri-Netze zur Extraktion von Steuerflüssen aus den Datenabhängigkeiten im Datenfluss), wie es für die vorliegende Arbeit in [Ko13] untersucht wurde.

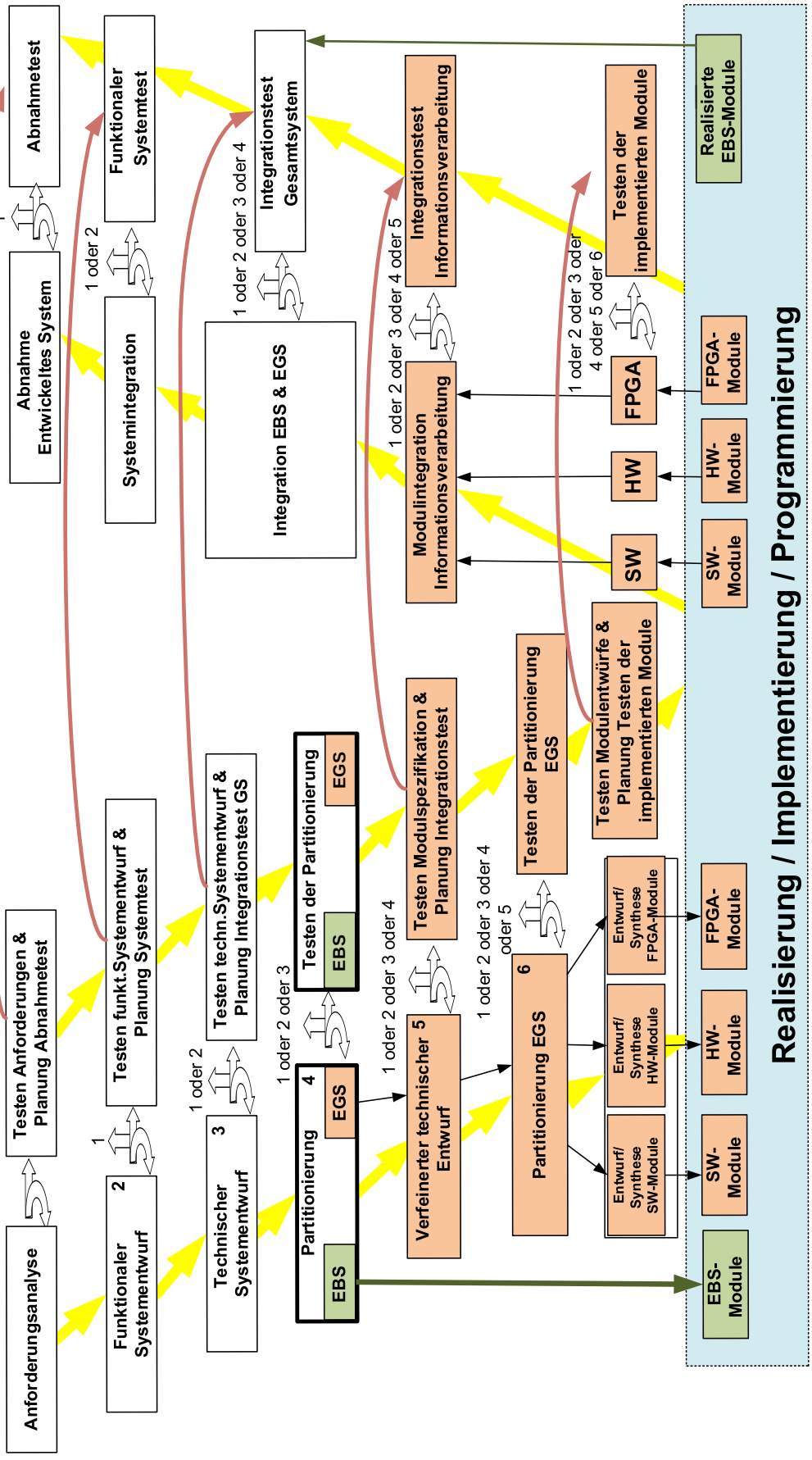


Abbildung 3.2: W-Modell der Entwicklung eines Prototyps

Für diese Dissertation werden Eingebettete Systeme mit rekonfigurierbarer Hardware (in Form von FPGAs) als Zielplattformen für die Realisierung von Informationsverarbeitungsaufgaben vorgeschlagen. Von der Vielzahl an Vorteilen, die im Abschnitt 2.4.3.2 beschrieben wurden, spielt die Rekonfiguration von FPGAs eine wichtige Rolle bei der Entwicklung von physischen Prototypen. Es ist effektiv, sehr leistungsfähige und kostenaufwendige FPGAs zu verwenden. Es können möglichst viele Lösungsvarianten unterschiedlicher Komplexität, noch nicht optimierter Hardwarestrukturen zusammen mit Softwarelösungen (über Softcore-Prozessoren, eventuell erweitert um externe Prozessoren) untersucht werden. FPGAs bieten den Zugang zu parallelen und hochparallelen Lösungsvarianten bei der Prototypenentwicklung. In einigen Fällen kann es sinnvoll sein, anstelle der FPGAs beziehungsweise in Ergänzung zu diesen leistungsfähige softwarebasierte Plattformen einzusetzen.

Die Modulspezifikation für FPGA-Lösungen benötigt eine Partitionierung in herkömmliche Hardware, rekonfigurierbare Hardware und Software. Das gilt analog für die Teile Einbettendes System (Messsystem) und Eingebettetes System (Informationsverarbeitung). Die Partitionierungsschritte müssen nicht zwingend auf der gleichen Abstraktionsebene erfolgen. Typischerweise liegt die Partitionierung zwischen Einbettenden und Eingebetteten Systemen auf einer höheren Abstraktionsebene als die Trennung in Hard- und Software. Im dargestellten Konzept findet die Partitionierung (Abb. 3.2- vierter Schritt) zwischen Einbettenden (EBS) und Eingebetteten (EGS) Systemen vor dem verfeinerten technischen Entwurf des Informationsverarbeitungssystems statt. Die Partitionierung für das Eingebettete System (EGS) trennt die Entwicklung von Software-, herkömmliche Hardware- und FPGA-Komponenten. Im Weiteren werden bis zur Integration in das Gesamtsystem alle partitionierten Teile weitgehend getrennt entwickelt und getestet. Dabei bestehen weiter Anhängigkeiten zwischen den getrennten Entwicklungslinien. Bei den gefundenen Fehlern, die in Hierarchieebenen oberhalb der Partitionierung beziehungsweise bei der Partitionierung entstanden sind, erfolgt die Korrektur dort. Es ist typisch, dass sich nach der Partitionierung die einzelnen Teillösungen im Sinne von Abhängigkeiten beeinflussen. Zum Beispiel kann eine Veränderung des mechanischen Systems (EBS) geänderte Algorithmen der Signalverarbeitung bewirken, wobei sich die Teilsysteme des Eingebetteten Systems ändern. Unter Umständen ist eine Veränderung aller Partitionierungen sinnvoll oder gar notwendig.

Im Konzept ZEfIRA ist es wichtig die systematische Beschreibung (Dokumentation) jedes Entwurfsschrittes für die evolutionäre Prototypenentwicklung durchzuführen, so dass diese für zukünftige Entwicklungsgenerationen als Vorläufersystem nutzbar wird.

In dieser Arbeit wird die prototypische Entwicklung eines Informationsverarbeitungssystems für die messtechnische Anwendung genutzt. Bei dieser ist es möglich, unterschiedliche Varianten von Algorithmen, Implementierungsvarianten und Zielplattformen der

produzierbaren Lösung für verschiedene Waagentypen zu untersuchen. Dabei können unterschiedliche, flexible Partitionierungsmöglichkeiten und Zielplattformlösungen betrachtet werden. Im Kapitel 4 werden die detaillierten Entwicklungsschritte, vorwiegend des linken W-Top-Down-Vorgehens, entwickelt und dargestellt. Exemplarisch werden die Entwicklungen komplexerer Wägesysteme als praktischer Hintergrund genutzt.

3.2 W-Prozess eines Vorläufersystems

Der Entwurf des Messsystems und der Informationsverarbeitungslösung beeinflussen sich gegenseitig, wobei sehr häufig bestimmte Teillösungen des Systems mit speziellen Anforderungen zum Beginn der Prototypenentwicklung nicht bekannt sind oder nicht berücksichtigt werden. Aus diesem Grund ist es günstig, ein funktionierendes Vorläufersystem – als Mustersystem für die Entwicklung zu nutzen (z.B. Nanomessmaschine 1 als Vorläufersystem für die Nanopositionier- und Messmaschine NPMM-200 [Am10]). Das ist besonders sinnvoll, wenn das Vorläufersystem auf dieselbe Art und Weise modellbasiert und mit Hilfe von Prototypen entwickelt wurde. Dieses ist aber nicht immer der Fall.

Obwohl das Mustersystem ein fertiges, funktionierendes System darstellt, werden bei dieser Variante alle Entwicklungsetappen in ein modifiziertes W-Modell (Abb. 3.3) überführt. Das Vorläufersystem unterscheidet sich typischerweise in seinen Parametern und Funktionen. Es werden hier Anforderungen, Spezifikationen und Designentscheidungen von diesem analysiert, vor allem auf Basis von vorhandenen Dokumenten. Diese Analyse und eine eventuelle nachträgliche Dokumentationsergänzung werden sinnvollerweise in einem Prozess durchlaufen, der an die W-Modelle des Prototyps und der produzierbaren Lösung angelehnt ist. Entscheidend ist, dass die benötigten Kenntnisse auf denselben Entwurfsetappen übernommen werden, wodurch die Entwicklung des geplanten neuen Systems in den zwei genannten Varianten (Prototyp und Produkt) effektiver wird.

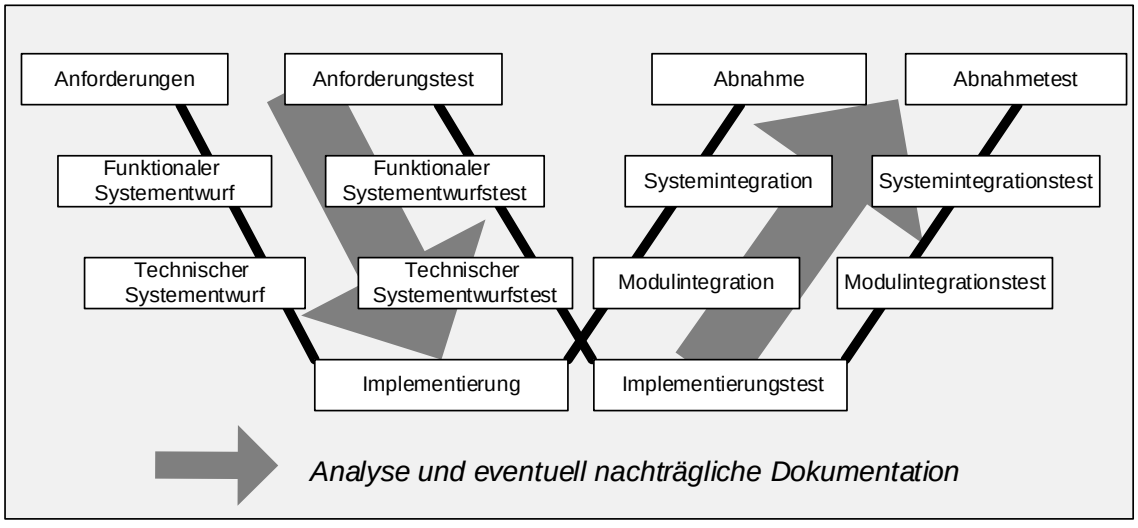


Abbildung 3.3: W-Modell eines Vorläufersystems (Prototyp und Produkt)

Dieses W-Modell wird vor allem mit dem W-Modell der Prototyp-Entwicklung gekoppelt, um Vor-/Nachteile sowie Lösungen des alten Systems im neuen Entwicklungsprozess zu berücksichtigen. Die beiden Modelle stehen zweifach in Wechselwirkung zueinander: Ein Prototyp wird auf Grund von neuen und alten Kenntnissen entwickelt und das Vorläufersystem kann letztlich durch die neuen Lösungen des Nachfolgersystems modernisiert werden, falls ersteres weiter produziert und vermarktet werden soll.

Unter der Modernisierung wird in diesem Fall kein grundsätzlich neues Produkt verstanden, sondern nur begrenzte Verbesserungen technischer und wirtschaftlicher Parameter durch Modifikationen desselben (z.B. veränderte Softwarevarianten). Das W-Modell aus der Abbildung 3.3 wird deshalb, wie die Abbildung 3.4 zeigt, ergänzt. Dabei werden erprobte Lösungsvarianten aus der neuen Prototypenentwicklung typischerweise auf den niedrigeren Entwurfsebenen übernommen. Sinnvoll ist eine Dokumentation der Auswirkungen in den höheren Ebenen im Sinne des Bottom-Up-Entwurfs.

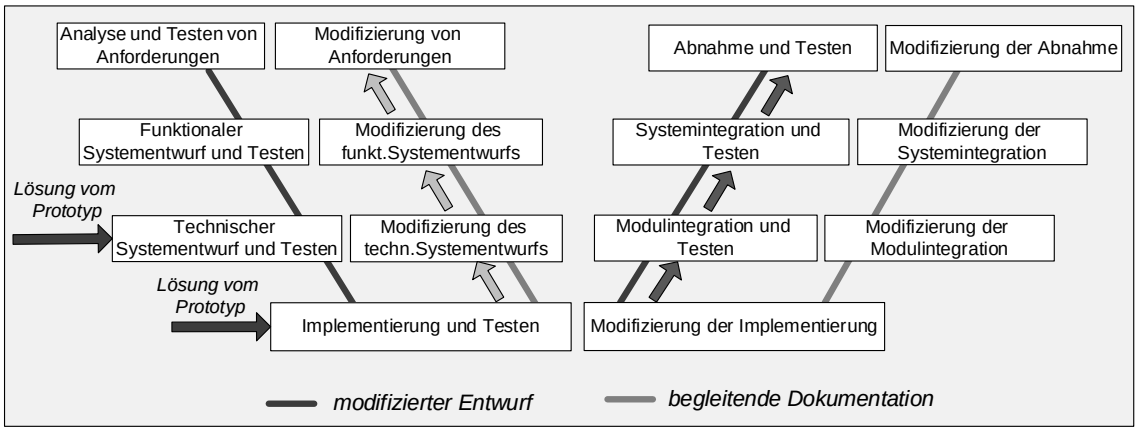


Abbildung 3.4: W-Modell der modifizierten Entwicklung im Vorläufersystem

3.3 W-Prozess einer produzierbaren Lösung

Das Konzept des 3W-Modells enthält einen dritten W-basierten Prozess zur Serienproduktentwicklung der entstandenen Lösung für die Informationsverarbeitung. Dieser beschreibt die Etappen der Entwicklung bei der Verwendung von anwendungsspezifischen Zielplattformen für die Produktion in relevanten Stückzahlen. Hierbei sind systemoptimierte und leistungsangepasste FPGAs, bei hoher Stückzahl die kostengünstigeren ASICs, oder auch kombinierte Plattformen mit DSPs, Mikroprozessoren beziehungsweise Universalprozessoren gemeint.

Das dritte W-Modell ist mit den zwei anderen W-Prozessen gekoppelt und wird zusammen mit diesen betrachtet. Die Prozesse der Prototypenentwicklung und der Produktenwicklung werden zeitversetzt abgearbeitet.

Absehbare Anforderungen (z.B. Einschränkungen der realisierbaren Funktionen bei der Verwendung von anderen Zielplattformen, andere Leistungsparameter) werden in allen Etappen der Produktentwicklung bei der Prototypenentwicklung berücksichtigt. Außerdem wird vor allem in der Wägetechnik ein Nachweis verlangt, dass ein Produkt die entsprechenden Gesetze der Eichbehörde, Normen, Richtlinien und Standards erfüllt (s. Abschnitt 2.2). Diese stellen verbindliche Anforderungen an die Messgeräte sowie deren Nachweis und Einhaltung während der Entwicklung und des Betriebs dar. Im Konzept ZEfIRA wird es möglich die verwendeten Entwicklungsmethoden und – werkzeuge für die Akzeptanz und die Zulassung durch entsprechende Organisationen in Prototypen zu berücksichtigen, selbst wenn noch keine normativen Regeln zu diesen vorliegen, aber mit einer zu erwartenden Wahrscheinlichkeit gesetzlich geregelt werden.

Die Abbildung 3.5 stellt das W-Modell der Produktentwicklung dar. Die Entwicklungsschritte von der Anforderungsanalyse bis zur Abnahme können vollständig bei der Prototypenentwicklung berücksichtigt werden. Aus diesem Grund sind diese direkt mit dem W-Modell des Prototyps gekoppelt. Beim Testen können Abweichungen abhängig von den Zielplattformen entstehen, wobei die Testverfahren und die Methoden der Prototypenentwicklung nur rudimentär wiederverwendet werden.

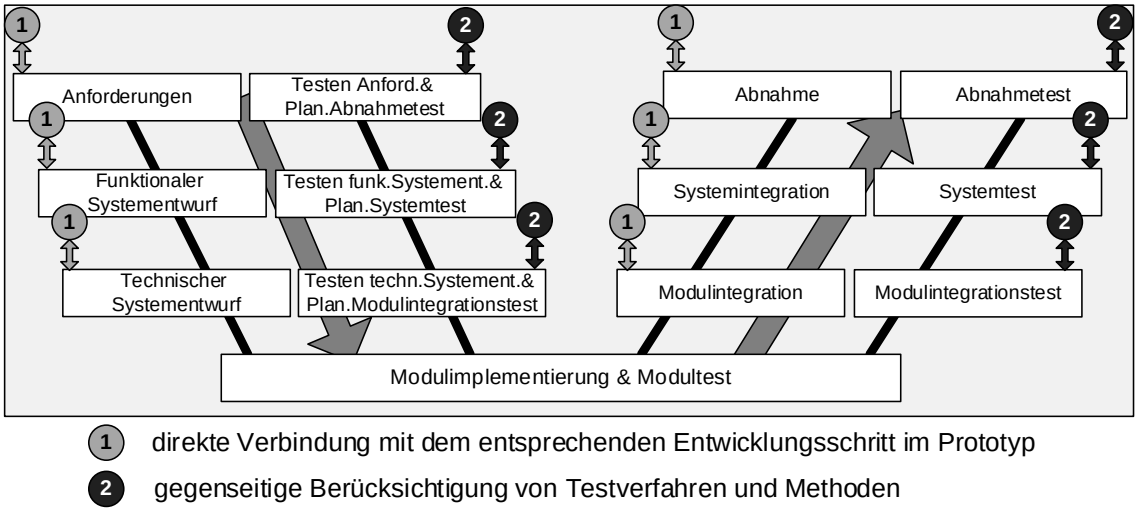


Abbildung 3.5: W-Modell der Produktentwicklung

Bei der Entscheidung der Modifikation des Vorläufersystems wird der dritte W-Prozess nicht berücksichtigt. Das bedeutet, dass dieser Prozess nur bei der Entwicklung des neuen Produkts im Konzept ZEfIRA aktiv abläuft.

3.4 Fazit zum 3W-Modell

Der entwickelte Prozess, der durch das 3W-Modell systematisiert und formalisiert wird, erhöht die Effektivität des Entwurfes und der Tests komplexer Informationsverarbeitungseinrichtungen in Hard- und Software in der Domäne der Messtechnik, beginnend mit einem eventuellen Vorläufersystem über einen Prototyp bis zu einem Produkt. Die Basis bildet ein modellbasiertes Top-Down-Vorgehen beim Prototypentwurf mit frühen Testaktivitäten bereits in der Entwurfszeit.

Der Prozess wird durch vorbereitende Testaktivitäten bei der Integration der entstandenen Lösungen nach dem Bottom-Up-Verfahren ergänzt. In allen Entwurfs- und Testetappen werden die gesetzlichen Anforderungen (z.B. Eichfähigkeit) berücksichtigt. Durch die Kopplung der Analyse des Vorläufersystems und der Entwicklung des Prototyps und des Produkts entstehen folgende Effekte:

- ✓ systematische Übernahme von Lösungen und Erfahrungen des Vorläufersystems;
- ✓ Flexibilität bei der Untersuchung von Lösungsvarianten im Prototyp;
- ✓ frühere und systematischere Erkennung und Korrektur von Fehlern und damit Erhöhung der Qualität des Entwicklungsergebnisses und der Metrologischen Sicherheit;
- ✓ direkter Übergang zu einem kostengünstigen Produkt auf Basis der günstigsten Prototypenlösung;
- ✓ Rückwirkung zur Verbesserung des Vorläufersystems, sofern dieses weiter produziert werden soll;
- ✓ Effektivitätssteigerung der Entwicklung und Verkürzung der Entwicklungszeiten;
- ✓ Zertifizierbarkeit des Prozesses, welcher zu einem zertifizierbaren Produkt führt;
- ✓ Geringerer Aufwand für den Nachweis der Eichfähigkeit bei den Eichbehörden.

Das vorgestellte Prozessmodell beschreibt den Rahmen, in dem die einzelnen Etappen integriert und vernetzt sind. Im Kapitel 4 werden die Prozessschritte einer effektiven Prototypenentwicklung detaillierter beschriebenen, wobei die Beziehungen zwischen den Etappen in den 3W-Modellen untereinander genauer betrachtet werden.

3.5 Eichfähigkeit und Metrologische Sicherheit in ZEfIRA

Der in dieser Arbeit entwickelte und dargestellte Entwicklungsprozess für Informationsverarbeitungssysteme hat mehrere Besonderheiten, die in der messtechnischen Domäne und insbesondere in der Wägetechnik zu berücksichtigen sind. Die speziellen Anforderungen in der Messtechnik, bezogen auf das Zeit-, Leistung- und Genauigkeitsverhalten sowie die Anforderungen an die Eichfähigkeit und zusätzlich die geeigneten Maßnahmen

zur Zertifizierung verhalten sich gegensinnig zu den Entwicklungs- und späteren Gerätekosten der Lösung. Diese Systeme verlangen die Einhaltung von speziellen messtechnischen Kriterien: eingegrenzte Messunsicherheit und Metrologische Sicherheit. Letztere wird in der messtechnischen Literatur [BS10] nicht direkt beschrieben. Deswegen werden für diese Arbeit eine konkrete Definition des Begriffes „Metrologische Sicherheit“ und dessen Kriterien bezogen auf die Informationsverarbeitung in messtechnischen Systemen vorgeschlagen und im Abschnitt 3.5.1 beschrieben. Weiterhin werden Methoden zur Validierung und Verifikation dargestellt, die die Metrologische Sicherheit in der genannten Domäne erhöhen sollen.

3.5.1 Definition und Kriterien

Die Messunsicherheit, die im Abschnitt 2.1 als eine wichtige messtechnische Größe beschrieben wurde, soll bei der Entwicklung von Geräten ständig nachgewiesen werden. Es existiert für viele Messgeräteklassen eine Eichpflicht, die Maßnahmen zur regelmäßigen Überprüfung der Einhaltung der Messunsicherheit im Betrieb verlangt. Durch die Zulassung der Messgeräte beziehungsweise Messsysteme wird ein ordnungsgemäßer Betrieb sichergestellt.

Die komplexe Informationsverarbeitung in solchen Systemen benötigt während der Entwicklung den Nachweis, dass die Messunsicherheit des Messverfahrens durch sie gewährleistet wird. Die Einhaltung qualitativer und quantitativer Kriterien soll die Metrologische Sicherheit garantieren, die für die Arbeit wie folgt definiert wird.

Definition:

> *Die Metrologische Sicherheit in einem Informationsverarbeitungssystem eines Messgerätes ist die Gewährleistung der korrekten Funktion unter Berücksichtigung der geforderten Messunsicherheit über den gesamten Messbereich und unter allen zugelassenen Messbedingungen der messtechnischen Einrichtung für die Dauer der projektierten beziehungsweise gesetzlich zugelassenen Betriebszeit. Letzteres gilt für Messgeräte, für die eine Eichpflicht besteht.*

Die Gewährleistung der Metrologischen Sicherheit kann naturgemäß nicht mit hundertprozentiger Wahrscheinlichkeit realisiert werden. Das resultiert insbesondere daraus, dass mehrere qualitative Kriterien berücksichtigt werden. Die Hard- und Softwaresysteme des Messsystems tragen zu der Metrologischen Sicherheit bei. Für die folgenden Untersuchungen wird gefordert und realisiert, dass in allen Entwurfsetappen deren Nachweis berücksichtigt wird. Das beginnt in den höheren Abstraktionsebenen und wird systematisch in den niedrigeren Ebenen weiter geführt.

Auf Grund der neu eingeführten Definition soll in diesem Kontext die Eichfähigkeit eines Informationsverarbeitungssystems erläutert werden. Die Metrologische Sicherheit und

die zusätzlichen geeigneten Maßnahmen zur Manipulationssicherheit sollen die Eichfähigkeit sowie die Einhaltung der im Vorrangegangenen beschriebenen Standards, Vorschriften, Richtlinien und Ähnlichem sicherstellen.

Die Metrologische Sicherheit ist eines der wesentlichen Ziele der Entwicklung eines Messgerätes. Sie muss immer im Kontext der technischen und wirtschaftlichen Anforderungen gesehen und berücksichtigt werden, wobei vor allem das Folgende erfüllt werden muss:

- ✓ Berücksichtigung von Eichfähigkeitsanforderungen an das Messgerät, welches aus dem Messsystem und der Informationsverarbeitung besteht (hier als Schwerpunkt der vorliegenden Untersuchung);
- ✓ Berücksichtigung von Anforderungen über die Eichfähigkeit hinaus (Standards, Vorschriften, Richtlinien und Ähnliches);
- ✓ Verwendung eines systematischen Entwicklungsprozesses, wodurch die Fehlerwahrscheinlichkeit sinkt;
- ✓ modellbasierte Entwicklung und modellbasiertes Testen;
- ✓ Verifikation und Validierung von allen Entwicklungsschritten und Teilprozessen mit geeigneten Methoden;
- ✓ Betrachtung der systematischen und zufälligen Fehler zum Nachweis der Metrologischen Sicherheit (Fehleranalyse jedes Entwurfsschrittes, mathematische Modellierung, Berechnung des Beitrages jedes Messgliedes).

3.5.2 Berücksichtigung der Metrologischen Sicherheit in der Messkette

Die Abbildung 3.6 stellt die Messkette mit Funktionsblöcken der Informationsverarbeitung dar, die typischerweise durch ein Eingebettetes System realisiert wird. Die Beiträge zur Metrologischen Sicherheit werden diesen dort zugeordnet. Die dazu zu betrachtenden Aspekte und Maßnahmen bezüglich jedes Teilsystems müssen in den Entwicklungsprozess integriert werden. Die Metrologische Sicherheit eines Informationsverarbeitungssystems bezieht sich auf die Gesamtheit der einzelnen Aspekte. Außerdem gehören zur Gewährleistung der Metrologischen Sicherheit die allgemeinen Forderungen an das Messsystem (s. Abb. 3.6).

Im Kapitel 4 werden die entwickelten Methoden zur Berücksichtigung der Metrologischen Sicherheit als Bestandteil der Betrachtung der Lösungen bei der Prototypenentwicklung dargestellt. Ein wichtiger Gesichtspunkt ist die systematische Überprüfung (Validierung und Verifikation) jedes Entwurfsschrittes mit geeigneten Ansätzen. Diese wurden für die vorliegende Dissertation untersucht und umgesetzt.

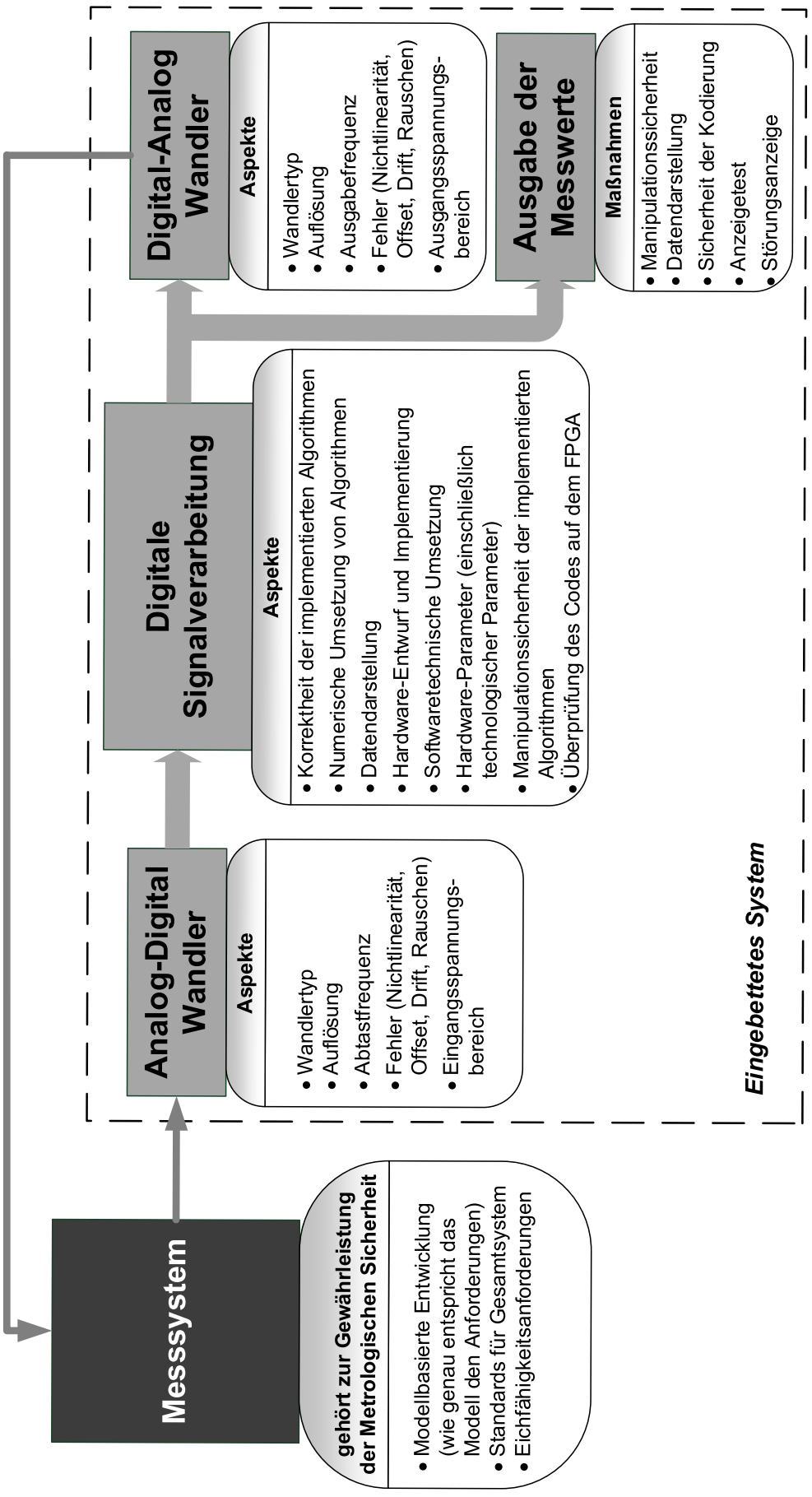


Abbildung 3.6: Betrachtung der Metrologischen Sicherheit in der Messkette

4 Prozessschritte der effektiven Prototypenentwicklung

In diesem Kapitel werden die Entwurfsetappen eines Prototyps beschrieben, der auf Basis des Entwicklungsprozesses ZEfIRA realisiert wird. Dabei stellt das Konzept des 3W-Prozesses die Implementierung eines evolutionären Prototyps in den Mittelpunkt. Dieser dient im Wesentlichen, wie im Kapitel 3 schon beschrieben wurde, der Entwicklung und Evaluierung von Algorithmen, Implementierungsvarianten und dem Einbettenden System. Dieses ermöglicht die Untersuchung von verschiedenen Lösungsmöglichkeiten und soll zu einer wirtschaftlich und technisch optimalen Implementierungslösung in der Produktenentwicklung führen.

Die vorliegende Dissertation beschäftigt sich vor allem mit der Entwicklung dieses evolutionären Prototyps für die Informationsverarbeitung. Die Entwicklungsschritte von der Anforderungsanalyse bis zur Implementierung stellen die Etappen des modellbasierten Entwurfs und deren Vorgehensweisen dar und werden hier vorzugsweise betrachtet. Als Applikationsbeispiele dienen Teile FPGA-basierter Eingebetteter Systeme für Informationsverarbeitungsaufgaben in der Messtechnik. Anhand der Applikation werden die Schritte im Weiteren näher erläutert. In diesem Zusammenhang wird das Eingebettete System (Informationsverarbeitungssystem) abhängig von den Forderungen des Einbettenden Systems (Messsystem) modellbasiert entwickelt. Dabei entstehen regelmäßig veränderliche Anforderungen an die Informationsverarbeitung auf Grund der vorhandenen Lösungsvariabilität in beiden Teilen (Eingebettetes und Einbettendes System). Es werden unterschiedliche Partitionierungen zwischen diesen und innerhalb der Informationsverarbeitung in verschiedenen Hierarchieebenen durchgeführt.

Bei der Entwicklung des Prototyps werden die folgenden hierarchisch aufeinander aufbauenden Abstraktionsebenen und Aktivitäten benötigt:

1. **Werkzeugunterstützte Anforderungsanalyse und Testen der Anforderungen.** Dabei ist die Berücksichtigung sowohl von speziellen Anforderungen an die Metrologische Sicherheit und die Eichfähigkeit als auch von Anforderungen der Produktenentwicklung nötig.

2. **Modellbasierter und werkzeugunterstützter funktionaler Systementwurf und dazugehörige Validierungs- und Verifikationsaktivitäten.** Hierzu steht das Eingebettete System in Wechselwirkung mit dem Einbettenden System.

3. **Modellbasierter und werkzeugunterstützter technischer Systementwurf mit dazugehörigen Validierungs- und Verifikationsaktivitäten.** Dabei werden typischerweise mehrere hierarchische Abstraktionsebenen benutzt. Es erfolgt eine Übernahme von eventuell modifizierten Modellen aus dem Vorläufersystem, falls diese vorhanden sind und die Übernahme sinnvoll ist. Danach folgt die iterative Partitionierung zwischen dem Einbettenden und dem Eingebetteten Systemen.

4. **Modellbasierter und werkzeugunterstützter verfeinerter technischer Entwurf des Eingebetteten Systems mit dazugehörigen Validierungs- und Verifikationsaktivitäten.** Es erfolgt, ähnlich wie bei dem Systementwurf, eine Übernahme von eventuell modifizierten Algorithmen und Prinzipien aus dem Vorläufersystem, falls diese vorhanden sind und diese Übernahme sinnvoll ist. Bei der Modellierung, Untersuchung und eventuellen Modifikation von Algorithmen wird eine Kombination von Daten- und Steuerfluss benutzt.

5. **Struktureller Entwurf, Validierung/Verifikation und Untersuchung von unterschiedlichen Implementierungsvarianten des Informationsverarbeitungssystems auf dem FPGA-basierten Eingebetteten System mit eventuellen weiteren Komponenten.** Dabei ist die iterative Trennung in die Partitionen Hardware, Software und FPGA notwendig. Die Aktion „hierarchischer Entwurf und Übernahme der vorhandenen Lösungen“ erfolgt ähnlich wie in den vorangegangenen Schritten. Weiterhin werden schon hier Restriktionen (z.B. Zeit, Ressourcen, Leistungsaufnahme) berücksichtigt.

6. **Getrennte Entwicklung von Modulen für Teilfunktionen in den Partitionen mit dazugehörigen Validierungs- und Verifikationsaktivitäten.** Auch hier kann eine Übernahme von vorhandenen Lösungen (Modulen) erfolgen. Wenn bei Validierungs-/Verifikationsaktivitäten Entwurfsfehler auftreten, werden diese in der Abstraktionsebene untersucht und behoben, in denen ihre Ursache liegt. Für Teile des Entwurfs werden deshalb die hierarchisch darunter liegenden Schritte erneut durchlaufen.

Im Weiteren wird eine detaillierte Beschreibung zum Ablauf des Entwicklungsprozesses von der Anforderungsanalyse bis zur Implementierung gegeben.

4.1 Anforderungsanalyse mit Validierung und Verifikation

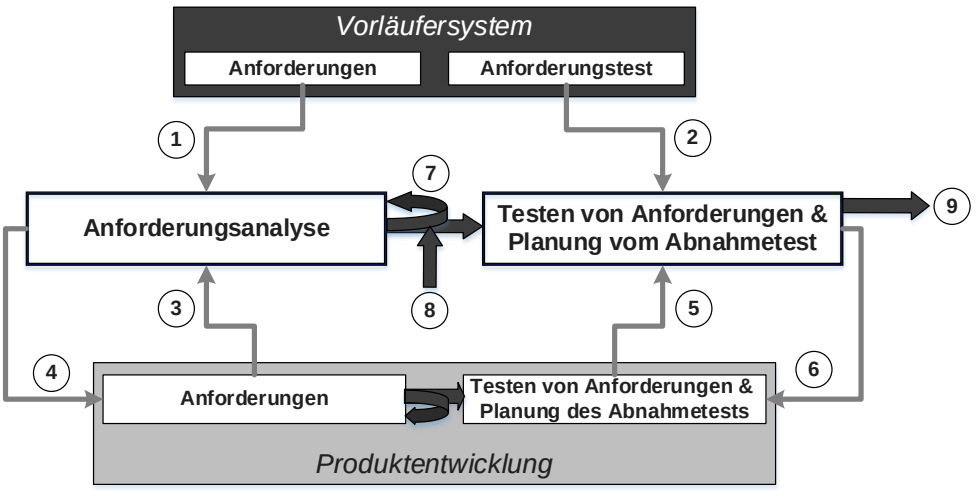
Die Betrachtung der Anforderungsanalyse vervollständigt den entwickelten W-Prozess in den obersten Hierarchieebenen. Die Untersuchungen dienen vorwiegend zur vollständigen Entwicklung dieses Prozesses. Sie sind nicht der Schwerpunkt der Arbeiten dieser Dissertation, beinhalten aber ein eigenes Konzept zur Durchgängigkeit von der Anforderungsanalyse bis zur Spezifikation und Implementierung.

Eine gute Anforderungsermittlung ist Voraussetzung für eine weitgehend fehlerfreie Entwicklung. Durch das Testen in früheren Phasen können nicht nur fehlerhafte Anforderungen korrigiert, sondern auch ungenügend oder gar nicht definierte Anforderungen bestimmt werden. Nach der Norm IEEE 830 vom Institute of Electrical and Electronics Engineers gibt es acht Qualitätskriterien für Anforderungsspezifikationen: Korrektheit, Vollständigkeit, Konsistenz, Prüfbarkeit, Eindeutigkeit, Verfolgbarkeit, Bewertbarkeit

und Modifizierbarkeit [R07]. Das Zusammentragen der Anforderungen kann auf unterschiedliche Art und Weise erfolgen. Dafür gibt es verschiedene, aus dem Bereich „Requirements-Engineering“ bekannte, Ermittlungstechniken sowie eine Vielzahl von Umsetzungsvarianten [FFH02].

Dieser Teilabschnitt widmet sich der Beschreibung der konzeptionellen Anforderungsanalyse für die Informationsverarbeitungsaufgaben in messtechnischen Systemen. Die Analyse soll aus verschiedenen Blinkwinkeln durchgeführt werden. Die Betrachtungswinkel können beispielweise Basisanforderungen, Systemanforderungen, Leistungsanforderungen, Qualitätsanforderungen und gesetzliche Anforderungen sein. Im Prozess ZEfIRA werden dazu die Anforderungen aus dem Vorläufersystem und die speziellen Anforderungen der Produktenentwicklung berücksichtigt.

Die Abbildung 4.1 stellt aus dem W-Prozess „Entwicklung eines Prototyps“ die Etappe für die Anforderungsanalyse mit den begleitenden Validierungs- und Verifikationsaktivitäten dar. In diesem Entwicklungsschritt entstehen die ersten direkten Kopplungen mit den weiteren zwei W-Prozessen (zum Vorläufersystem und zur Produktentwicklung). Außerdem sieht man, dass gegenseitige Einflüsse zwischen der Prototypenentwicklung und der Produktentwicklung existieren, obwohl nicht alle Testfälle vollständig berücksichtigt beziehungsweise übernommen werden können.



1. Berücksichtigung von dokumentierten Anforderungen aus dem Vorläufersystem
2. Mögliche Übernahme von Testfällen und -verfahren aus dem Vorläufersystem
3. Berücksichtigung von spezifischen Anforderungen der Produktentwicklung inklusive gesetzliche Anforderungen
4. Anpassung von Anforderungen der Produktentwicklung abhängig von den Ergebnissen der Analyse im Prototyp
5. Untersuchung von bei der Produktentwicklung notwendigen Anforderungstests im Prototyp
6. Teilweise Wiederverwendung von Testverfahren und Methoden vom Prototyp
7. Iterative Überprüfung, Präzisierung und Anpassung von Anforderungen im Prototyp
8. Präzisierung von Anforderungen auf Basis von Informationen aus hierarchisch untergeordneten Ebenen
9. Übernahme von Testfällen für den Abnahmetest

Abbildung 4.1: Anforderungsanalyse im Prototyp

Zur Beherrschung der Komplexität der modellbasierten Darstellung und Untersuchung müssen an dieser Stelle die allgemeinen Methoden der Anforderungsanalyse verwendet werden. Die im Requirements-Engineering bekannte zielbasierte Anforderungsmodellierung wird in dieser Dissertation als ein passendes Konzept gewählt. Darauf basierend müssen die folgenden Schritte durchgeführt werden:

- Unterscheidung zwischen Zielen und Anforderungen bei der Analyse,
- Ableitung von Anforderungen aus den Zielen,
- Schrittweise Verfeinerung von Anforderungen zur Beherrschung der Komplexität,
- Operationalisierung von Anforderungen durch überprüfbare Kriterien,
- Unterteilung in funktionale und nichtfunktionale Anforderungen.

Die Abbildung 4.2 zeigt eine vereinfachte Gesamtdarstellung für die Ermittlung von Anforderungen mit deren Ableitung aus den Zielen. In diesem Zusammenhang sind die wesentlichen Methoden dazu angegeben. Der Prozess der Ableitung der Anforderungen aus den Zielen kann formalisiert und überprüfbar organisiert werden [R07]. Um Anforderungen in einer gewissen Weise testen zu können, sollen sie in eine teilformale Beschreibung überführt werden.

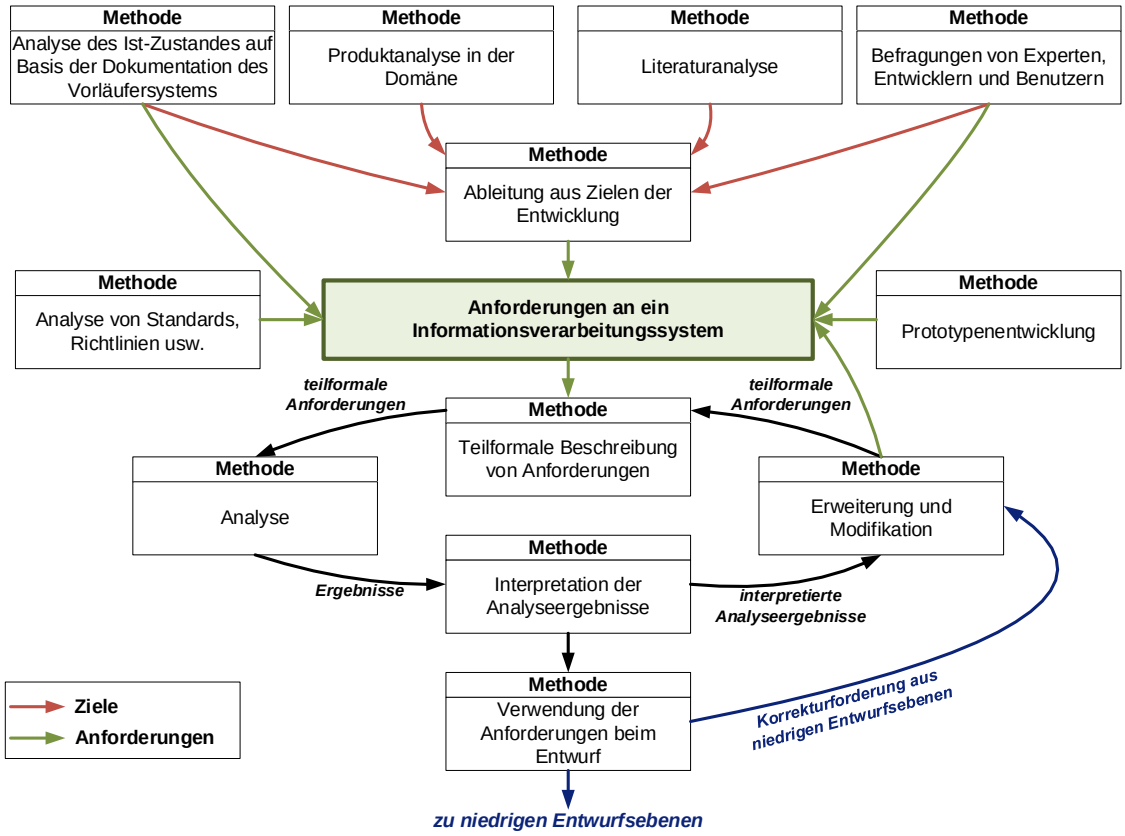


Abbildung 4.2: Methoden zur Ermittlung von Anforderungen

Die Unterteilung in funktionale und nichtfunktionale Anforderungen ist eine wichtige Aufgabe für die Systementwicklung [R07]. Dabei beschreiben die funktionalen Anforderungen die Fähigkeiten des Systems, die benötigt werden, um mit Hilfe dieses Systems ein Problem zu lösen. Als Beschreibungsmittel werden häufig Anwendungsfälle (engl. *Use Cases*) verwendet. Die nichtfunktionalen Anforderungen beschreiben Anforderungen an das System, die nicht direkt die oben genannten Fähigkeiten betreffen (vor allem Sicherheits-, Qualitäts-, gesetzliche Anforderungen). Dabei definieren diese die grundlegenden Eigenschaften des Systems, die als Restriktionen beziehungsweise Nebenbedingungen beim Entwurf berücksichtigt werden müssen. Es ist notwendig, die nichtfunktionalen Anforderungen, soweit möglich, messbar zu beschreiben. [Mae09]

In der Masterarbeit von Frau Weichenhain [Wei12] wurden einige für die vorliegende Dissertation interessante Ansätze für Requirements-Engineering und Projektmanagement in der Wägetechnik erarbeitet und beschrieben. Ersteres wird im Weiteren näher betrachtet. Im Rahmen der Arbeit wurde ein grober Überblick zur Ermittlung von Anforderungen geschaffen, welche eine eigene Untersuchung und eine Verfeinerung benötigen. Der dazu aufgestellte Lebenszyklus einer Anforderung soll bei der korrekten Behandlung behilflich sein. Weiterhin muss eine Priorisierung der aufgestellten Anforderungen erfolgen, weil nicht alle die gleiche Bedeutung für die konkrete Anwendung besitzen.

Ein wichtiger Punkt ist das Testen von Anforderungen. Es dient deren qualitativer Verbesserung bei der Untersuchung und stellt ein Instrument der Qualitätssicherung dar. Die Analyse erzielt eine genaue und detaillierte Beschreibung sowie die Prüfung der Vollständigkeit und Exaktheit. Sie verbessert die Verständlichkeit der Anforderungen. Nicht alle von ihnen können präzise vorhersehbar sein, wodurch Unsicherheiten auftreten (z.B. Einfluss von Störgrößen). Aus diesem Grund ist es notwendig die Toleranzanforderungen dementsprechend anzupassen.

Die Abbildung 4.3 stellt den durch Frau Weichenhain erarbeiteten und für diese Dissertation erweiterten Lebenszyklus einer Anforderung dar. Der Lebenszyklus beginnt mit der Zieldefinition und der Ableitung einer Anforderung aus den Zielen. Diese resultieren in der betrachteten Anwendungsdomäne aus der dynamischen Wägetechnik und den für ein konkretes Erzeugnis festgelegten Parametern und nichtfunktionalen Produkteigenschaften. Abgeleitete Anforderungen müssen anschließend definiert und formuliert werden. Dabei sind Eindeutigkeit und Testbarkeit besonders zu beachten.

Die folgende Bewertung hat das Ziel, den Anforderungen unterschiedliche Prioritäten zuzuweisen. Dieses erleichtert die nachfolgende Umsetzung unter deren Berücksichtigung. Wenn die Anforderung positiv beurteilt wird, kann das Definieren von Testfällen mit anschließenden Bewertungskriterien vorgenommen werden. Die Durchführung enthält Aktivitäten zur Umsetzung und zum Testen. Negative Testergebnisse stoßen einen

iterativen Prozess zur Fehlersuche an. Die erfolgreiche Durchführung der Anforderungsanalyse ist dann gegeben, wenn mindestens alle Anforderungen für die Erfüllung der Aufgabe (mit mittlerer und hoher Priorität) erfolgreich umgesetzt werden konnten.

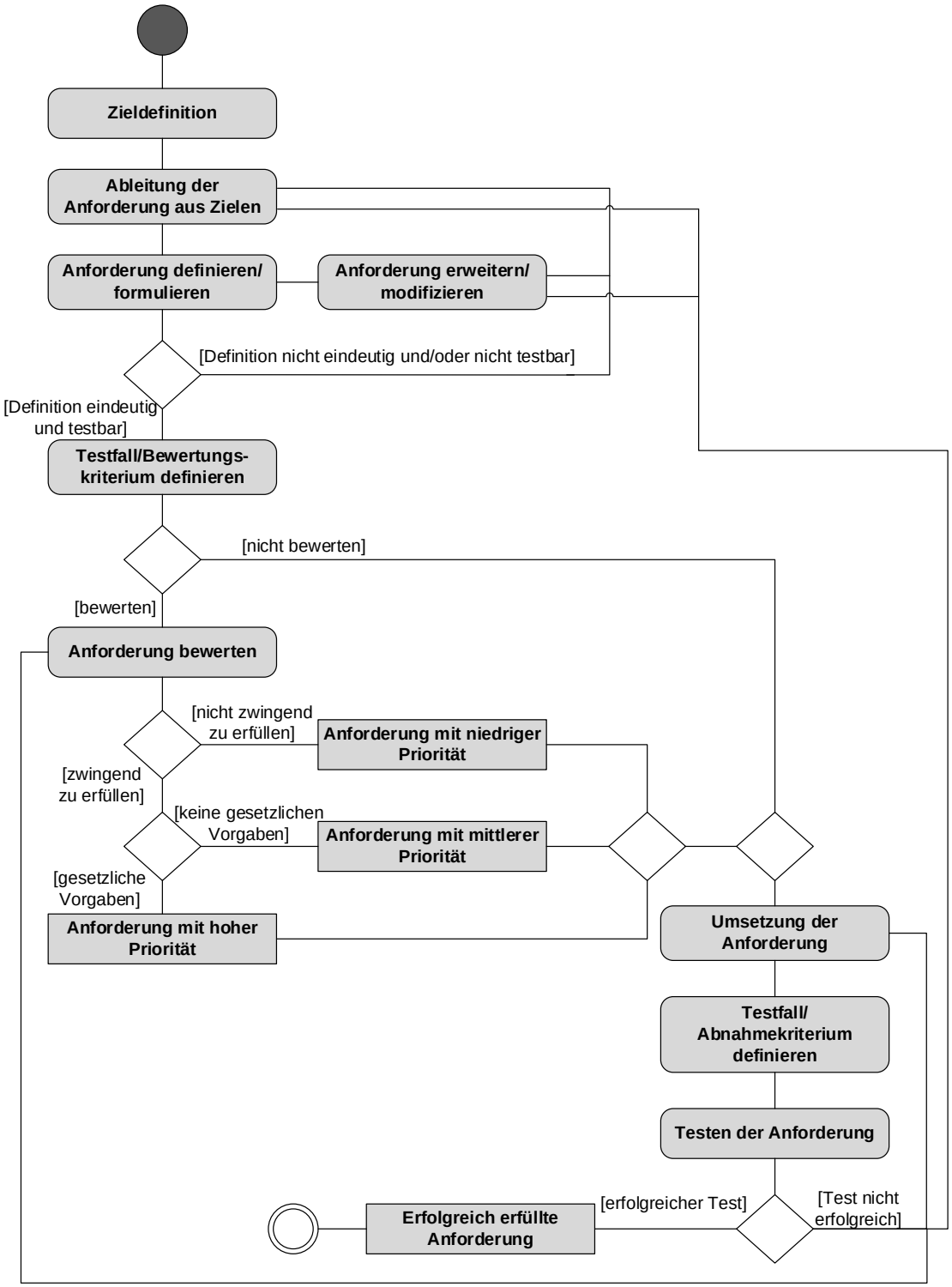


Abbildung 4.3: Lebenszyklus einer Anforderung

In dem Forschungsprojekt [Ch14] wurde die Verwendung eines speziell für den Entwurf in der Messtechnik zugeschnittenen Werkzeuges bezüglich unterschiedlicher Techniken zur Analyse und zum Testen, beginnend mit den Anforderungen bis zur Spezifikation, bei der Entwicklung eines Informationsverarbeitungssystems, konzipiert und exemplarisch umgesetzt. In Hinblick auf die vollständige Unterstützung aller Entwicklungsetappen (von der Anforderungsanalyse bis zur Implementierung einschließlich Verifikations- und Validierungsaktivitäten) gibt es zurzeit kein geschlossenes Werkzeug. In diesem Zusammenhang muss man sich auf die Integration von professionellen Systemen für die Anforderungsanalyse (z.B. Dynamic Object Oriented Requirements System - DOORS) in den gesamten Entwicklungsprozess orientieren. Für diese Dissertation wurde ein anderes Verfahren verfolgt. Durch die Verwendung eines in der Entwicklungsumgebung zur Verfügung gestellten kommerziellen Werkzeugs (z.B. NI Requirements Gateway in LabVIEW) kann diese Integration automatisiert werden. Im Abschnitt 5.2.1 wird ein exemplarisches Beispiel der Anforderungsanalyse in der dynamischen Wägetechnik bei der Verwendung des NI Requirements Gateway dargestellt und das eben genannte Konzept und dessen Umsetzung genauer beschrieben.

4.2 Funktionaler Systementwurf mit Validierung und Verifikation

Der funktionale Entwurf ist ein iterativer Prozess, in dem auf Basis der Anforderungen eine stabile grobe Architektur des zu entwickelnden Gesamtsystems und der möglichen beziehungsweise nötigen Umgebungssysteme definiert wird. Dabei werden die Anforderungen aus der im Abschnitt 4.1 beschriebenen Anforderungsanalyse übernommen und geeignet aufbereitet. Die entsprechenden Anforderungen werden den Architekturkomponenten zugeordnet, zu denen sie in Beziehung stehen. Abschließend wird das System mit seinen Schnittstellen auf hohem Abstraktionsniveau entwickelt und beschrieben. Die formalen Verifikationsaktivitäten ermöglichen Überprüfung von Eigenschaften des abgebildeten Gesamtsystems. Voraussetzung ist deren formale Beschreibung. Bei Vorhandensein einer ausführbaren Gesamtsystemspezifikation können Funktionalitäten auch hier schon durch Simulation, d.h. durch eine Testmethode (Validierung), untersucht werden [Bau09]. Es wird im Rahmen der Anforderungsverfolgung [Mae09] sichergestellt, dass alle vorher definierten Anforderungen berücksichtigt und umgesetzt werden können. Hier besteht eine bidirektionale Verfolgbarkeit. Dadurch ist die Verfeinerung und Detaillierung von Anforderungen möglich.

Die Abbildung 4.4 stellt die Etappe des funktionalen Systementwurfes im W-Prozess des Prototyps dar. Dieser Entwurf beinhaltet die Verfeinerung und die Überprüfung der weitgehenden Korrektheit und technischen Realisierbarkeit der Ergebnisse.

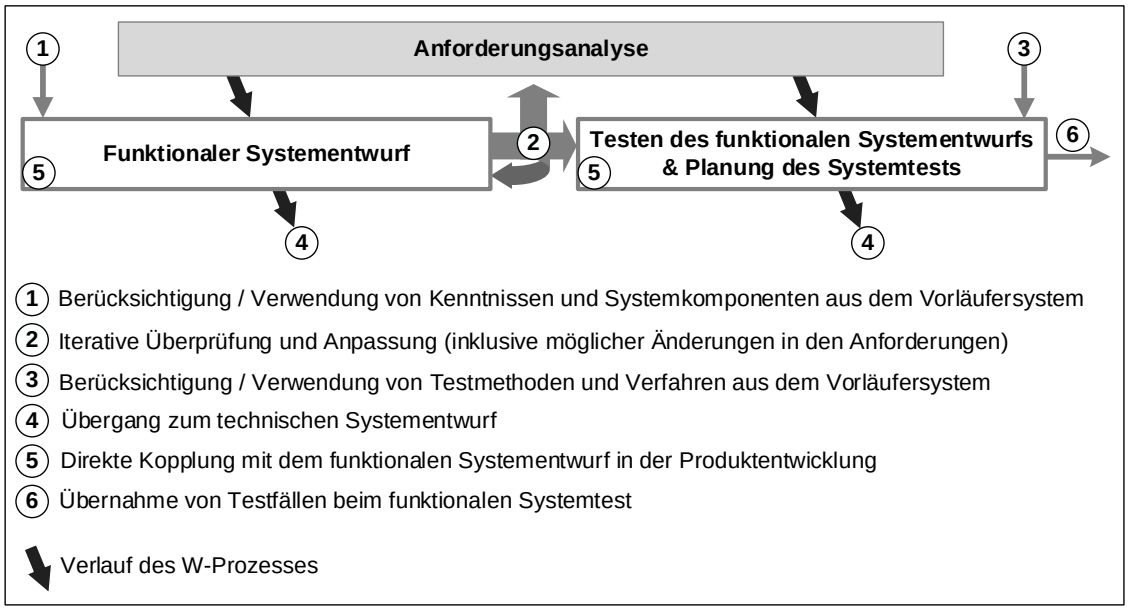


Abbildung 4.4: Etappe des funktionalen Systementwurfs

Die Beschreibung von komplexen Informationsverarbeitungssystemen und dazugehörigen Aufgaben würde ohne hierarchisch aufeinander aufbauende Abstraktionsebenen zu einer schwer überschaubaren Darstellung führen. Deshalb sollen die umfangreichen Komponenten (Aufgaben) in mehrere Teilkomponenten (Teilaufgaben) abhängig von den funktionalen Aspekten des Systems und deren inhaltlichem Zusammenhang unterteilt werden. Bei der Form des modellbasierten funktionalen Systementwurfs werden die Komponenten blockorientiert dargestellt. Die Verbindungen dieser Blöcke beschreiben den Daten- und Steuerfluss, letzteren über Steuersignale und Zeitangaben. Zeiten können auch Blöcken direkt zugeordnet werden. Hierzu werden die Schnittstellen der Blöcke verbunden. Es ist sinnvoll, dass bei der Modellierung der Systemstruktur (Blöcke und deren Verbindungen) bereits getroffene Partitionierungsentscheidungen frühzeitig berücksichtigt werden.

Zur Beschreibung von modellierten Komponenten beziehungsweise Teilkomponenten wird eine Analyse des Verhaltens durchgeführt. Für das statische Verhalten werden die Struktur, das heißt vor allem die Schnittstellen, und die Funktionalität eines Systemelements festgelegt. Es gibt dafür unterschiedliche Möglichkeiten: formal (z.B. Gleichungen), teilformal (z.B. Fachbegriffe), informal (z.B. textuell). Inhalte und Beschreibung von Systemkomponenten können abhängig von Bereichen des Einbettenden und Eingebetteten Systems variieren. Elektronische beziehungsweise mechanische Bereiche werden durch die Angabe von elektrischen beziehungsweise mechanischen Daten spezifiziert, Bereiche der Informationsverarbeitung durch die Beschreibung von Methoden, Parametern und Informationen zum Verhalten. Das dynamische Verhalten bestimmt die Reihenfolge der Nutzung und die logischen Abhängigkeiten der übermittelten Informationen (Signale in der Messkette).

Die verwendeten Werkzeuge beziehungsweise Entwicklungsumgebungen für diese Etappe sollen aus den bisher genannten Gründen Untersuchungen beziehungsweise Entwürfe auf unterschiedlichen Abstraktionsebenen ermöglichen. Die dabei nutzbaren Validierungs- und Verifikationsaktivitäten sollen sowohl über die Hierarchieniveaus der betrachteten Entwurfsetappe hinausreichen als auch in Verbindung zur Anforderungsanalyse, zum technischen Systementwurf und zu eventuell darunterliegenden Ebenen stehen. In diesem Zusammenhang ist die Unterstützung durch integrierte Werkzeuge notwendig.

Im Prozess ZEfIRA orientieren sich die Darstellung und die Beschreibung von Komponenten unter anderem auf die Wiederverwendung bereits bestehender Systemelemente beziehungsweise Kenntnisse und Informationen aus dem Vorläufersystem. Auf Basis von Dokumentationen können Grundlagen der Architekturübersicht sowie der Spezifikationsrealisierung der Systemelemente angewendet werden. Darüber hinaus ist bei der Dokumentierung der Lösungen des aktuell zu entwickelten Prototyps darauf zu achten, dass alle Systemkomponenten so umfangreich beschrieben und stabil sind, dass damit eine effektive Wiederverwendung der Informationen in weiteren Generationen (Prototypen und Produkten) möglich wird.

Als Beispiel wird ein in der vorliegenden Dissertation betrachtetes und entwickeltes Wägesystem mit digitaler Informationsverarbeitung graphisch dargestellt.

Die Abbildung 4.5 zeigt die Komponenten der Messkette, die beim funktionalen Systementwurf beschrieben, verifiziert und validiert wurden (s. Beispiele im Kapitel 5).

Die Beschreibung des statischen und dynamischen Verhaltens für jede Komponente der Messkette erfolgt über die folgenden Angaben:

- **Messsystem (hier EMK-Wägezelle):** textuelle Beschreibung der Funktion bezogen auf die Schnittstellen (Auslenkung *Y*, Kompensationskraft *F*, Gewichtskraft *Fg*), mechanische Auslegung, technischer Aufbau, Hinweise zur Benutzung und Bedienung der Schnittstellen (Ein- und Ausgänge).
- **Elektronik einschließlich Elektromechanik** (hier Lageindikator, U/I-Wandler, Spule, I/U-Wandler): textuelle Beschreibung der Funktion bezogen auf die Schnittstellen (z.B. Lageindikator: Auslenkung *Y*, Positionsspannung *Upos*), elektrische Leistungsdaten (Leistung, Spannung, Strom, Widerstände, Frequenz, Polarität), technische Auslegung/Aufbau (Kabeltyp, Steckerbelegung), Beschreibung der Spezifikation (Benutzung und Bedienung).
- **Analoge Ein- und Ausgänge (AD- und DA-Wandler)**: technische Daten und Charakteristiken (Prinzip und Funktion bezogen auf die Schnittstellen (z.B. *Upos* als Spannung und digitaler Wert), Kanäle, Spannungs- und Stromkennwerte, Auflösung, Abtast- und Ausgabefrequenz, Fehler, Umwandlungsbereich), Beschreibung zur Benutzung und Bedienung von Schnittstellen.

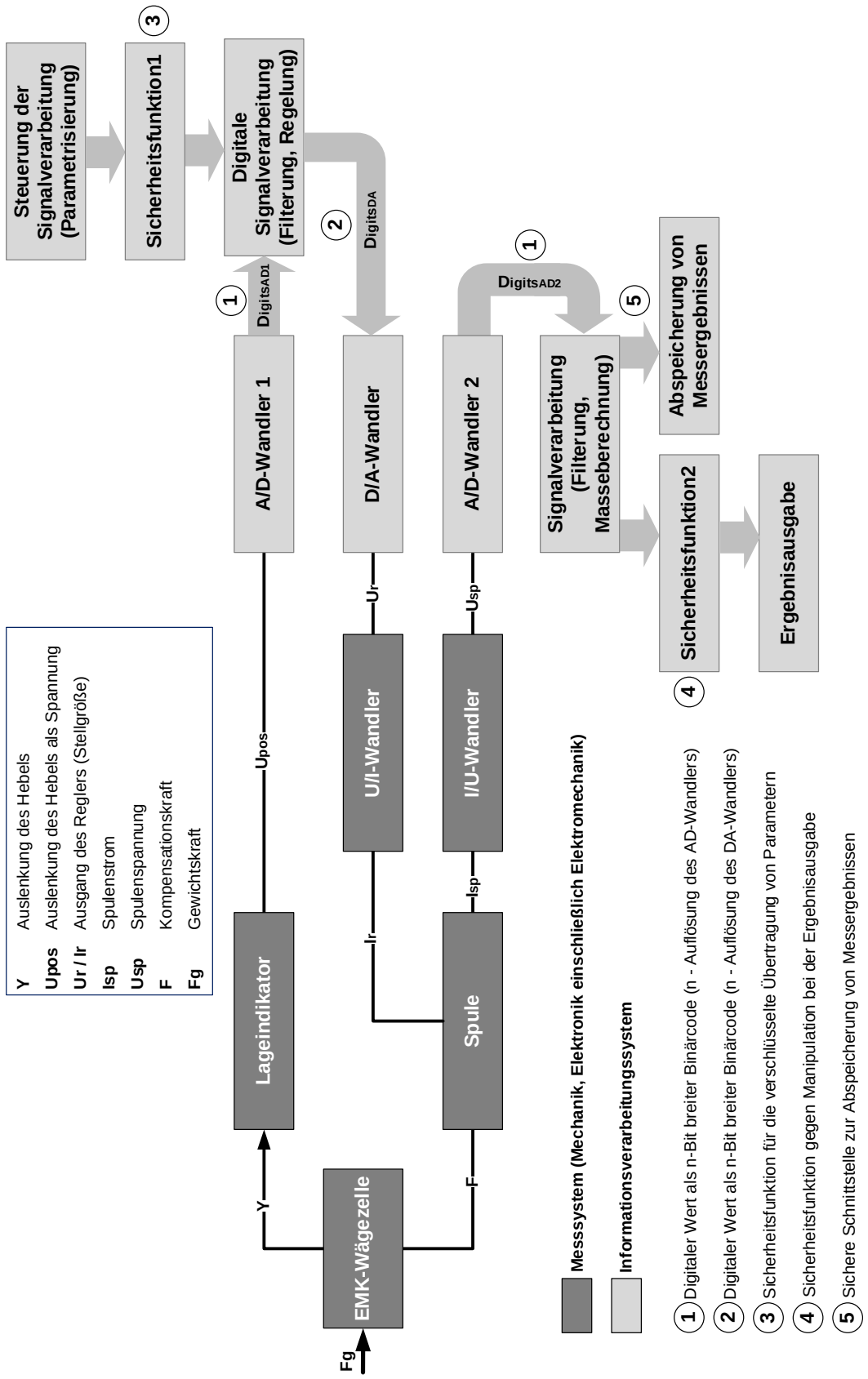


Abbildung 4.5: Komponenten der Messkette für ein EMK-Wägesystem

- **Komponenten der Informationsverarbeitung** (hier Signalverarbeitung, Steuerung der Signalverarbeitung, Sicherheitsfunktionen, Abspeicherung, Ausgabe): Beschreibung und Definition zu verwendeten Methoden (Algorithmen) und deren Schnittstellen (Kommunikationsprotokolle und deren Spezifikation), Strukturaufbau und dynamisches Verhalten (z.B. Scheduling der Blöcke und Verwendung von Parametern), Festlegung von Aufrufreihenfolgen (z.B. beschrieben durch Sequenzdiagramme, Statecharts).

Bei der Entwicklung von komplexen Informationsverarbeitungssystemen haben die Verfeinerungen der beschriebenen Systemkomponenten schon in dem Schritt des funktionalen Entwurfs eine große Bedeutung. In diesem Fall können strukturelle und funktionale Verfeinerungen durchgeführt werden.

Von besonderem Interesse sind in dieser Arbeit die Komponenten des Eingebetteten Systems. Im Weiteren werden ausgewählte Ergebnisse zu durchgeführten Untersuchungen dargestellt, die der Etappe funktionaler Systementwurf zuzuordnen sind.

4.2.1 Strukturelle Verfeinerung der Spezifikation

Eine strukturelle Verfeinerung einer Komponente (eines komplexen Blocks) ist die Unterteilung in Teilkomponenten mit entsprechenden Schnittstellen. In diesem Zusammenhang wird die Funktionalität der komplexen Komponente auf zugeordnete Teilkomponenten verteilt. Dadurch werden die Systeme mit wachsender Komplexität schrittweise beherrschbar und leichter verständlich. Außerdem können unterschiedliche Eigenschaften von Systemkomponenten bezüglich ihrer inneren Struktur zusätzlich analysiert werden. In diesem Fall nimmt die Menge von Aussagen zu Eigenschaften zu. Es können auch zusätzliche Eigenschaften dazu kommen.

In dem oben genannten Beispiel (s. Abb. 4.5) können die Blöcke der Signalverarbeitung strukturell verfeinert werden. Die Komponente „Digitale Signalverarbeitung“ lässt sich z.B. in zwei Bereiche unterteilen: Filterung und Regelung. Für jede Teilkomponente wird die Darstellung in der Regel formal mit der Beschreibung des Strukturaufbaus und des dynamischen Verhaltens erfolgen. In einem weiteren Schritt können diese Teile unabhängig voneinander tiefer verfeinert werden. Neben der Aufteilung in Teilkomponenten befasst sich die strukturelle Verfeinerung mit der internen Kommunikationsstruktur, das heißt mit Schnittstellen zwischen den Teilkomponenten und den Zuordnungen der internen Ein- und Ausgänge.

Eine wichtige Rolle spielt die Überprüfung von Verfeinerungen. Für alle Varianten sollen spezifische Regeln zur Verfügung stehen, mit denen Verfeinerungsrelationen formal nachgewiesen werden können. Als Prinzip gelten die folgenden Regeln:

- Verfeinerung A besitzt alle Methoden im Hinblick auf die syntaktischen Schnittstellen, die die oben angeordnete Komponente K besitzt;
- jede syntaktisch korrekte Spezifikation, die mit der Komponente K korrekt funktioniert, erlaubt die Ersetzung von K durch die Verfeinerung A, ohne die Korrektheit der Spezifikation zu zerstören;
- Das Schnittstellenverhalten der Verfeinerung A stimmt in Hinblick auf die Methoden der Komponente K mit dieser überein.

Bei der Verfeinerung im funktionalen Systementwurf ist es sinnvoll, unterschiedliche Restriktionen bezüglich der verfügbaren Elemente, Ressourcen, Konstruktionsregeln und Ausführungszeiten einzufügen und zu betrachten. In diesem Fall können die grundsätzlichen Begrenzungen oder Einschränkungen, die auf der Anforderungsanalyse, auf Kenntnissen aus dem Vorläufersystem oder auf speziellen Forderungen der produzierbaren Lösung basieren, schon in früheren Entwicklungsphasen analysiert werden.

Ein Beispiel der Verfeinerung eines IIR-Filters in eine Struktur enthält der Anhang A.1. Dabei existieren auf dem Niveau des funktionalen Entwurfs diese Abstraktionsebenen: informale Beschreibung als Block (IIR-Filter), Differenzialgleichung und Differenzengleichung. Letztere kann direkt in eine Struktur überführt werden, die die Gleichung direkt abbildet. Im Entwicklungssystem MATLAB/Simulink gibt es die Möglichkeit einen entsprechenden Filterblock direkt zu parametrisieren. Die integrierten Tools erzeugen daraus ohne Zwischenschritte des Entwicklers die strukturelle Beschreibung (s. Abschnitt 5.2.3.2: Realisierung eines Elliptischen Filters). Aus dieser kann rückwirkend die Differenzengleichung aufgestellt werden, um das funktionale Verhalten darzustellen. In diesem Fall handelt es sich um einen Buttom-Up-Entwurfsschritt.

Im Laufe der modellbasierten Entwicklung können mehrere Verfeinerungsaktivitäten auf unterschiedlichen Abstraktionsebenen erfolgen. Von besonderem Interesse für diese Arbeit sind der technische Systementwurf und darin die Verfeinerung am Übergang von einer plattformunabhängigen zu einer plattformspezifischen Darstellung. Die Abschnitte 4.3 und 4.4 widmen sich dieser Thematik detaillierter.

4.2.2 Modul zur Analyse von Analog-Digital-Wandlern

Zur Verfeinerung der funktionalen Spezifikation eines Blockes ist es sinnvoll, dessen global beschriebene Funktion genauer zu analysieren. Dieses geschieht im vorliegenden Teilabschnitt beispielhaft anhand der Analyse von Analog-Digital-Umsetzern.

Die Auswahl von für das konkrete Messsystem geeigneten analogen Ein- und Ausgängen der Informationsverarbeitung ist eine wichtige Aufgabe beim Systementwurf. Diese Komponenten tragen die Verantwortung für die Auflösung der abgetasteten und ausgegebenen Signale. Dabei wird vor allem die Messunsicherheit beeinflusst (Abschnitt 2.3).

Bei der Betrachtung der Metrologischen Sicherheit der Informationsverarbeitung sind die AD-Umsetzer die ersten Glieder der Messkette, die eine Untersuchung bezüglich der Anwendbarkeit und der Fehleranalyse erfordern. Die Besonderheit ist, dass im Ergebnis ein Bauelement auszuwählen ist, dessen Funktion im Allgemeinen nicht durch Algorithmen oder Implementierungsvarianten beeinflussbar ist. Im diesem Zusammenhang gibt es Ausnahmen, z.B. die Erhöhung der Auflösung durch Downsampling (Abschnitt 5.2.3.2).

In einem Forschungsprojekt, bearbeitet von Herrn Ravi Mambally D. [Mam14], wurde für diese Dissertation ein Modul zur Analyse und Auswahl eines passenden AD-Wandlers entworfen. Die Realisierung des Moduls erfolgte in MATLAB/Simulink, wobei dessen Programmierung direkt in der Sprache MATLAB durchgeführt wurde. Das ermöglicht eine Wiederverwendung in der Entwicklungsumgebung MATLAB/Simulink oder eine Integration in eine andere Umgebung (z.B. LabVIEW). Die Abbildung 4.6 stellt das realisierte Modul mit entsprechenden Ein- und Ausgängen dar.

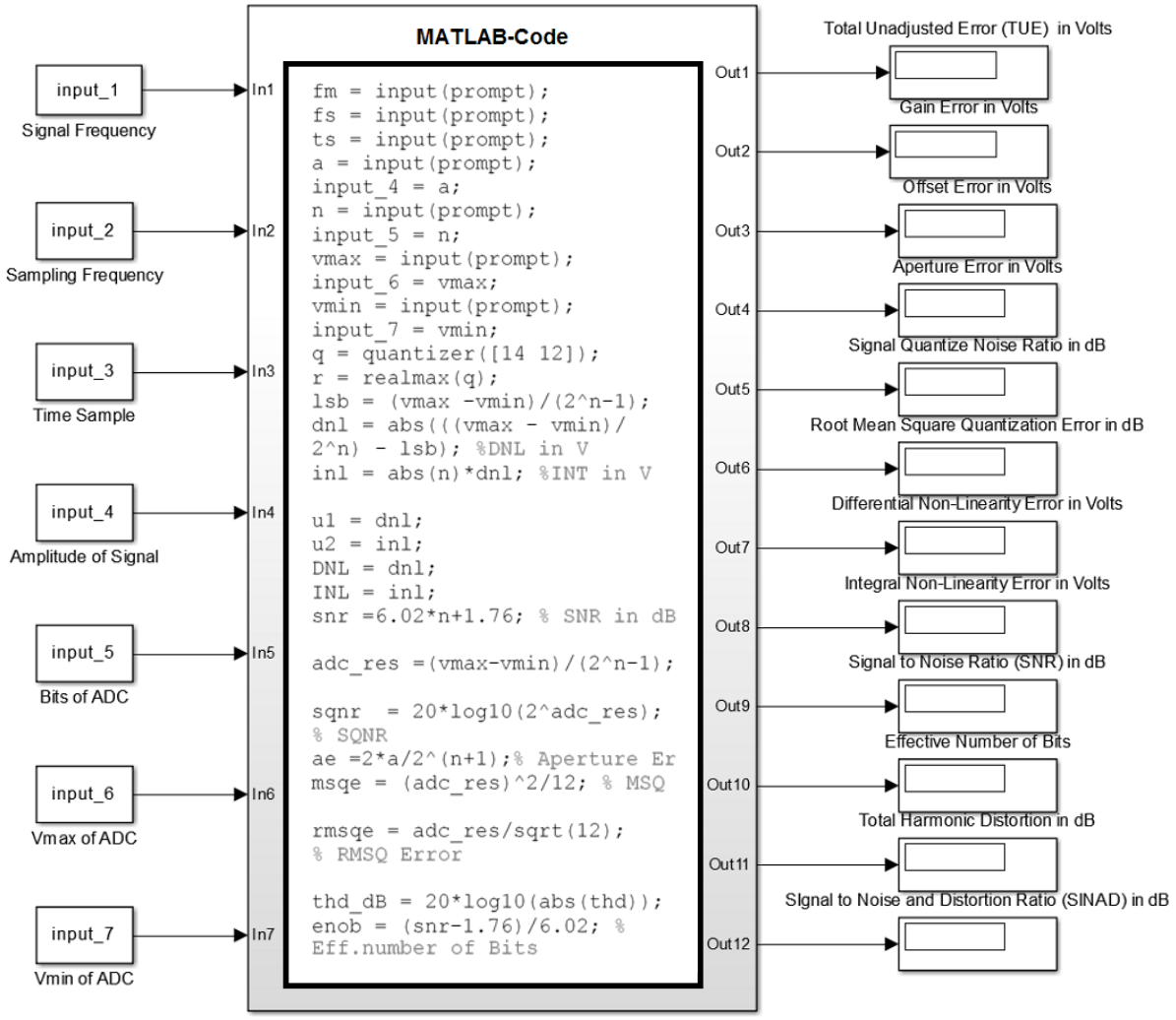


Abbildung 4.6: Modul zur Analyse von Analog-Digital-Wandlern

4.2.3 Funktionen für die Metrologische Sicherheit

Zur Gewährleistung der Metrologischen Sicherheit und der Erfüllung der gesetzlichen und anderen Anforderungen müssen im Informationsverarbeitungssystem unterschiedliche Funktionen umgesetzt werden. Außerdem wird in der Regel ein Schutz der Informationsverarbeitung gegen Angriffen implementiert. Es kann aber Möglichkeiten geben, diesen zu umgehen. Dieser Abschnitt gibt einen Überblick über Stellen in der Messkette, die eine besondere Betrachtung dieses Aspekts bei der Realisierung benötigen.

Die erste Gruppe von Sicherheitsfunktionen bezieht sich auf den Schutz gegen Manipulationen für die äußere Sicherheit. Beispiele sind in der Abbildung 4.5 gezeigt. Es muss verhindert werden, dass die zur Steuerung der Signalverarbeitung übertragenen Parametern (Sicherheitsfunktion 1) und/oder die Messergebnisse, hier die berechnete Masse (Sicherheitsfunktion 2), manipuliert werden können. In der Diplomarbeit von Bock [B12] wurden unterschiedliche Gegenmaßnahmen untersucht. Zu diesen gehören Checksummen, digitale Signaturen und Verschlüsselung. Letztere wurde auf Grund von Analysen als die am meisten verwendete und sicherste Art ermittelt. Die gesamte Information (Parameter, Ergebnisse) wird mittels eines kryptographischen Verfahrens verschlüsselt. Die Ver- und Entschlüsselung sind hier nur mit einem korrekten Schlüssel möglich.

Eine andere Gruppe von Funktionen wird gegen die internen Angriffe verwendet. Darunter wird vor allem verstanden, dass Fehlfunktionen, die von außen aktiviert werden oder die bei der Entwicklung entstanden sind, das Verhalten der Informationsverarbeitung modifizieren. Zu einen gehören dazu die nichtbeabsichtigen Fehler von Entwicklern und Entwicklungstools, die die Metrologische Sicherheit verringern. Dabei kann in beiden Fällen auch beabsichtigtes Fehlverhalten eine Rolle spielen. Als Beispiel in diesem Sinne kann der formale Nachweis der Korrektheit eines genutzten Open-Source Softcore-Prozessors genannt werden [Sch12]. Im Rahmen dieser Arbeit wurden Fehler gefunden, korrigiert und anschließend bewiesen, dass mit der Korrektur der Fehler eine vollständig richtige Funktion vorhanden war. Zu anderem hat insbesondere der Speicherschutz gegen diverse Manipulationen beziehungsweise unerwünschte Veränderungen eine wichtige Bedeutung. In einem Informationsverarbeitungssystem wird ein Speicher normalerweise zur Abspeicherung von unterschiedlichen Programmkonfigurationen und Messergebnissen gebraucht. Wenn die Speicherstruktur und die Speicherorte von Daten einem Angreifer bekannt sind, können sie manipuliert werden. Der Schutz sensibler Daten kann erhöht werden, indem z.B. eine Prüfsumme oder eine kryptografische Hashfunktion verwendet werden. Auf Grund der relativ einfachen Implementierbarkeit und der breiten Wirksamkeit hat diese Sicherheitsgruppe eine besondere Bedeutung in einem Informationsverarbeitungssystem.

In der vorliegenden Dissertation werden auch weitere zusätzliche Schutzmechanismen und Schutzmaßnahmen erarbeitet. Diese basieren nicht direkt auf den Anforderungen der

WELMEC-Standards, sind aber zur Gewährleistung der Metrologischen Sicherheit notwendig.

Die erste Maßnahme bezieht sich auf die Möglichkeit, die implementierten Funktionen (Algorithmen) überprüfen zu können. Dazu werden die Korrektheit und die Plausibilität der Funktionalität der Informationsverarbeitung nachgewiesen. Wenn eine Funktion manipuliert wurde oder keine sinnvollen Ergebnisse mehr liefert (z.B. wegen Ausfall eines Hardware-Blocks), muss dieses entdeckt, angezeigt und die Messwertausgabe unterdrückt werden. In der Arbeit wird eine derartige Sicherheits- und Prüfkomponente durch den Einsatz eines hierfür entwickelten Online-Testdatengenerators realisiert [MP14]. Dieser erzeugt Eingangsdaten für bestimmte Testobjekte (z.B. Funktionen der Signalverarbeitung) und überprüft die Ergebnisse mit den zugehörigen, vorher berechneten erwarteten Werten. Dabei läuft der Prozess parallel zum eigentlichen Programmablauf der Informationsverarbeitung.

Eine weitere Schutzmaßnahme bezieht sich auf die Überwachung von Ausführungszeiten. In diesem Zusammenhang wird eine Zeitberechnungs- und -auswertungskomponente realisiert, die die Zeit der Berechnung einer bestimmten Operation oder Funktion während der Ausführung mit einem erwarteten Wert vergleicht. Diese Realisierung ist damit auch während der Entwicklung zum Test der Einhaltung von zeitlichen Forderungen verwendbar.

Die obengenannten Funktionen wurden in der Arbeit exemplarisch umgesetzt. Sie sind in weiteren Abschnitten (Kapitel 5) detaillierter dargestellt. Dabei geht es um eine typische Auswahl, die in Abhängigkeit der konkreten Anwendung ergänzt oder modifiziert werden soll.

4.3 Technischer Systementwurf mit Validierung und Verifikation

Die Etappe des technischen Systementwurfes beschäftigt sich mit der Realisierung der Ergebnisse des funktionalen Systementwurfes. In diesem Zusammenhang wird das Gesamtsystem, beginnend mit der Architektur, detailliert und so präzise beschrieben, dass eine technische Umsetzung möglich wird. Zu weiteren Aspekten gehören z.B. verwendete oder für die technische Realisierung verwendbare Beschreibungsmittel, Toolauswahl und eingesetzte Technologien. Der technische Systementwurf wird häufig in Grob- und Feinentwurf zerlegt. Demzufolge wird eine Grob- beziehungsweise Feinarchitektur entworfen.

In diesem Abschnitt wird die Entwurfsetappe des Grobentwurfes mit nachfolgender Partitionierung beschrieben. Dabei wird die Methode der modellbasierten plattformunabhängigen Entwicklung verwendet. Als Ziel wird der modulbasierte werkzeugunterstützte Entwurf von Komponenten des Einbettenden und des Eingebetteten Systems verfolgt.

Die Abbildung 4.7 stellt die Etappen des technischen Systementwurfs mit der nachfolgenden Partitionierung entsprechen dem W-Modell des Prototyps dar. Die Kopplung mit dem W-Prozess der produzierbaren Lösung ermöglicht die frühzeitige Berücksichtigung von Kenntnissen über den späteren technischen Systementwurf in der Produktentwicklung. Anderseits wird der Einfluss des technischen Entwurfs des Prototyps aus dem des Produkts beachtet.

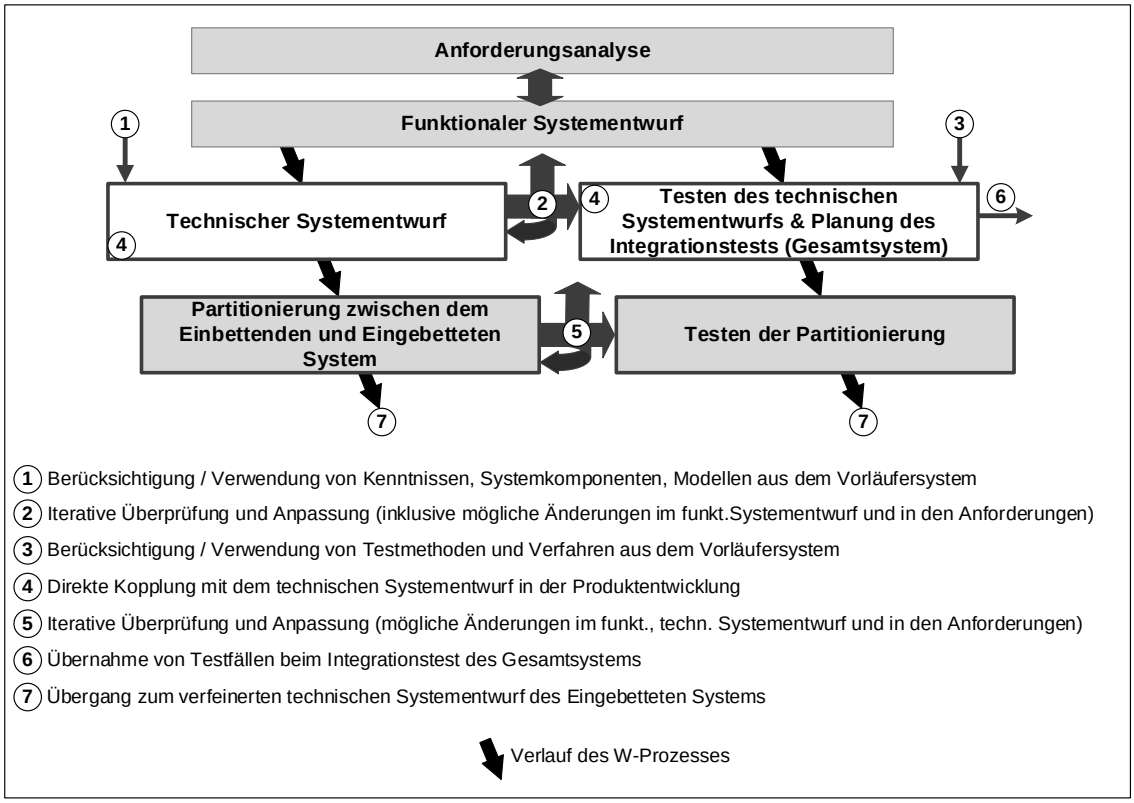


Abbildung 4.7: Etappe des technischen Systementwurfs

Der technische Systementwurf mit den begleitenden Verifikations- und Validierungsaktivitäten ist ein iterativer Prozess. Dieser enthält mehrere Verfeinerungsstufen. Beginnend mit der Berücksichtigung und der Wiederverwendung von Kenntnissen, Systemkomponenten und vorhandenen Modellen aus dem Vorläufersystem wird die erste vollständige Architektur auf Basis von Ergebnissen des funktionalen Systementwurfes erstellt. Weiterhin erfolgen Überprüfungen und Anpassungen von Systemkomponenten und deren Modellen. In diesem Zusammenhang ist es nicht auszuschließen, dass die zusätzlichen Änderungen und Anpassungen im funktionalen Systementwurf und/oder in der Etappe der Anforderungsanalyse durchgeführt werden müssen. Abschließend wird eine grobe

technisch realisierbare Struktur festgelegt, die in die Partitionen „Einbettendes“ und „Eingebettetes“ System geteilt wird. Diese Partitionierung benötigt eine Validierung, die einen gewissen Sicherheitsgrad bei der Festlegung von darauf abgebildeten Teilen des Gesamtsystems schafft. Anderseits kann eine nichtrealisierbare Struktur durch die Partitionierung entstanden sein. In diesem Fall können Änderungen in den technischen und funktionalen Systementwürfen, aber auch bei den Anforderungen nötig werden.

Zur Realisierung des Gesamtsystems werden in dieser Arbeit datenfluss- und steuerflussorientierte graphische Modelle zugrunde gelegt. Diese ermöglichen eine plattformunabhängige Beschreibung und dienen zur Realisierung von simulierbaren Berechnungsmodellen. Hierzu ist die Verwendung von kommerziellen Werkzeugen sinnvoll, wie z.B. MATLAB/Simulink und LabVIEW, die diese Entwurfsmechanismen unterstützen. In diesem Zusammenhang ist eine zunehmende Konkretisierung beziehungsweise Verfeinerung von Modellen und deren Eigenschaften erforderlich.

4.3.1 Modellbasierter Entwurf des Informationsverarbeitungssystems

Die vorliegende Dissertation beschäftigt sich schwerpunktmäßig mit dem Entwicklungsprozess von Informationsverarbeitungssystemen für die messtechnische Domäne. Dieses ist nur in einer engen Verbindung mit dem Einbettenden System möglich und verlangt die Erstellung von Modellen des Einbettenden und des Eingebetteten Systems.

Zur Modellierung und Untersuchung eines messtechnischen Systems gibt es bereits Arbeiten, z.B. die Dissertation von Frau Baumgartl [Ba15]. Diese Arbeit beschäftigt sich mit der Optimierung dynamischer Waagen nach dem Prinzip der elektromagnetischen Kraftkompensation mittels numerischer Modelle zur Systemsimulation. Die elektromechanischen Modelle wurden ausschließlich mit Signalverarbeitungsmodellen getestet, die nur auf dem technischen Entwurf des Vorläufersystems beruhten und nicht modellbasiert entwickelt wurden. Aufbauend auf diesen Ergebnissen wurden für die vorliegende Dissertation in Kooperation mit dem Fachgebiet Systemanalyse der TU Ilmenau neue Modelle des elektromechanischen Teils erarbeitet, die für die Entwicklung von leistungsfähigeren Signalverarbeitungsalgorithmen geeignet sind [AKRW14]. Die folgenden Abschnitte beschreiben diese sowie die wichtigsten erarbeiteten Modelle des Informationsverarbeitungssystems. Durch die Kopplung beider Teile konnte das Gesamtsystem simulativ analysiert und im Ergebnis der Untersuchungen optimiert werden. Auf diesem Niveau konnten die erzielbaren Leistungssteigerungen des Gesamtsystems bereits betrachtet und nachgewiesen werden. Dieses ist das Vorgehen der iterativen Überprüfung und Anpassung im W-Prozess (s. Abb. 4.7: **2**). Zur Planung des Integrationstests des Gesamtsystems können die dabei entstandenen Testfälle angepasst und genutzt werden.

4.3.1.1 Modell des messtechnischen Systems

Im Laufe der Entwicklung von messtechnischen Systemen werden unterschiedliche Simulationen zur Überprüfung von einzelnen Komponenten des Systems durchgeführt. Unbeachtet bleiben aber häufig das Zusammenwirken aller Komponenten und auftretende Wechselwirkungen zwischen Mechanik, Elektronik und Signalverarbeitung. Eine Analyse und Optimierung komplexer Messsysteme und die frühzeitige Anpassung der Komponenten einschließlich einer Untersuchung von Schwachstellen wird durch die zugrunde gelegte modellbasierte Entwicklung unterstützt. Dafür kommen grundsätzlich numerische und/oder analytische Methoden zum Einsatz. Der numerische Ansatz erfolgt zumeist durch Finite-Elemente- (FEM) oder Finite-Differenzen-Modelle (FDM), die einen hohen Detaillierungsgrad erreichen, aber eine unzureichende Parametrisierbarkeit besitzen. Der analytische Ansatz legt die Modellierung des Systems mittels Differentialgleichungen zugrunde. In diesem Fall werden die Modelle vollständig parametrisierbar. Die Komplexität von mechatronischen Systemen benötigt die Vereinigung von beiden Ansätzen, um die Vorteile beider Methoden zu nutzen. [Ba15]

Die Etappe des effektiven technischen Entwurfes erfordert die Umsetzung der folgenden Aufgaben:

1. verfeinerte Modellierung des Gesamtsystems auf Grund der im funktionalen Systementwurf beschriebenen Struktur;
2. Untersuchung von Umsetzungsmöglichkeiten zur Modellierung der ermittelten und geforderten Funktionen der Messkette;
3. Erstellung von detaillierten Modellen der Komponenten und deren Schnittstellen im Einbettenden System;
4. Erstellung von detaillierten Modellen der Komponenten und deren Schnittstellen im Eingebetteten System;
5. Modellierung von Schnittstellen zur Verbindung des Einbettenden und des Eingebetteten Systems;
6. im Informationsverarbeitungssystem: Analyse und Auswahl eines für die konkrete Anwendung geeigneten Algorithmus;
7. Analyse und Auswahl einer für die konkrete Applikation geeigneten und optimalen Realisierungsstruktur.

Das genaue Vorgehen soll am Beispiel der Modellierung einer EMK-Wägezelle im Zusammenhang mit dem modellbasiert entworfenen Regelungssystem erläutert werden. Ein anderes Bespiel der Modellierung eines komplexen mechatronischen Systems findet man in [Am10]. Darin ist die Modellbildung für die Trajektorienfolgeregelungen in Nanopositionier- und Nanomessmaschinen detailliert beschrieben.

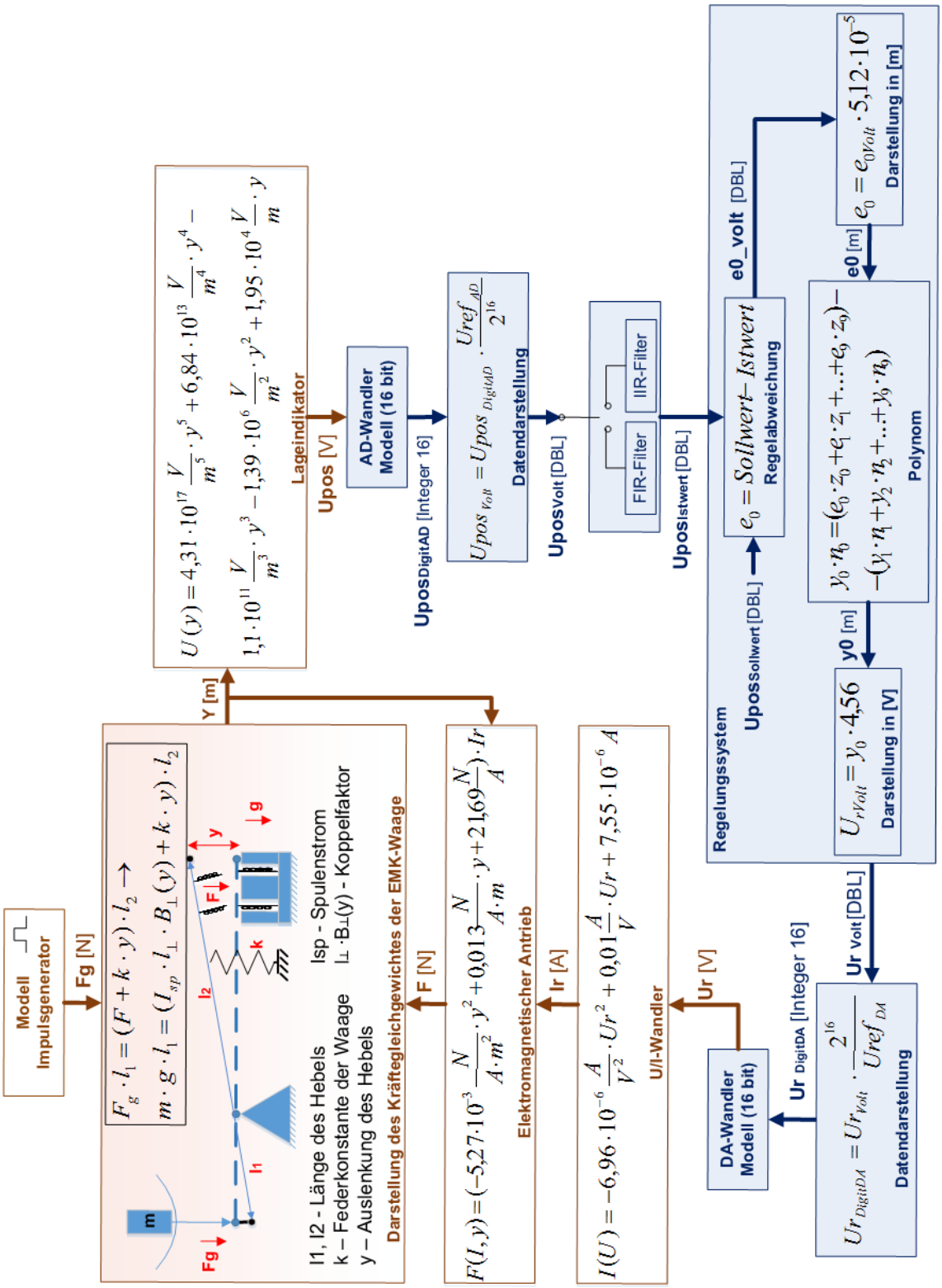


Abbildung 4.8: Modellierung des Gesamtsystems

Das modellierte Gesamtsystem ist in der Abbildung 4.8 gezeigt. Zur Realisierung des Systemmodells wurden das Einbettende und das Eingebettete System auf mehrere Komponenten verfeinert. Für die detailliertere Beschreibung der Komponenten des Einbettenden Systems ist eine genaue Identifizierung von deren Verhalten (z.B. Kennlinien, Parameter) durchzuführen. Deren Strukturen wurden bereits im funktionalen Systementwurf festgelegt. Während des weiteren Entwurfsvorganges kann sich herausstellen, dass diese Strukturen, Kennlinien und Parameter eventuell geändert werden müssen. Das entspricht den Iterationen aus dem technischen Systementwurf und der Partitionierung zurück zum funktionalen Entwurf und möglicherweise zu den Anforderungen (s. Abb. 4.7: **2** und **5**).

Die EMK-Wägezelle wird als ein Zwei-Balken-System durch Bewegungsdifferentialgleichungen beschrieben. Ausgehend von den Eingängen, der Kraft auf das Hebelende (F) und der Kraft auf die Waagschale (Fg) wird die Dynamik der Waage abgebildet. In der Masterarbeit von Goncharova [Go14] wurde für diese Dissertation ein vereinfachteres Modell in MATLAB/Simulink realisiert. Hierzu wurden unterschiedliche Varianten der Modellierung des messtechnischen Systems (mittels Übertragungsfunktionen und Zustandsraumdarstellung) untersucht. Der Systemausgang im mechanischen Wägezelle-Modell, die Position des Hebelarmes, wird durch numerische Integration ermittelt.

Die weiteren Komponenten des Einbettenden Systems, die modelltechnisch berücksichtigt und realisiert wurden, sind Lageindikator und Antriebsystem (Teile der Leistungselektronik (U/I-Wandler) und Spulenantrieb, beschrieben durch eine wegabhängige Kraftkonstante). Zur Bestimmung der physikalischen Größen jedes Übertragungsgliedes wurden die Kennlinien vermessen und identifiziert (s. Abb. 4.8). Dafür braucht man ein Verarbeitungssystem für die experimentelle Analyse des Messsystems und dessen charakteristischen Eigenschaften [AKRW14]. Das entspricht einer erweiterten Funktion des Prototyps zur Systemidentifikation, die im Kapitel 5 noch genauer beschrieben wird. Abhängig vom experimentellen Aufbau des Gesamtsystems soll eine geeignete Topologie für diese Systemidentifikation gewählt werden. Dafür wurden die Schnittstellen zur Umwandlung der Ein- und Ausgangsspannungen in die mechanischen Ein- und Ausgangsgrößen beschrieben.

Vom Eingebetteten System wurden Modelle für die regelungstechnischen Aufgaben realisiert, mit den Hauptkomponenten Analog-Digital-Wandler, Filterung, Regelung und Digital-Analog-Wandler. Außerdem wurden zur Umwandlung der unterschiedlichen Daten verschiedene Schnittstellenfunktionen zwischen diesen gebraucht.

Die Schnittstellen, die eine Verbindung zwischen den Funktionen der Signalverarbeitung und dem Einbettenden System darstellen, spielen bei der Entwicklung des Eingebetteten Systems eine wichtige Rolle. Es ist deren genauere Untersuchung und Modellierung notwendig, damit die nachfolgend gewünschten Funktionen mit einem angemessenen Rea-

litätsgrad für die konkrete Anwendung korrekt umgesetzt werden können. Eine Herausforderung stellt dabei die Modellierung von Analog-Digital- und Digital-Analog-Wandlern dar. Diese unterscheiden sich je nach Prinzip der Umsetzung und enthalten mehrere für die Analyse relevante Charakteristika, wie z.B. Abtastzeit, Auflösung sowie eine Reihe von Fehlerquellen (s. auch Abschnitt 4.2). In diesem Zusammenhang soll bei der Modellierung ein Kompromiss zwischen diesen Eigenschaften berücksichtigt werden. Auf diese Art wurde in der Bachelorarbeit von Pohl [P13] bei der Modellierung eines AD-Wandlers nach dem Prinzip der sukzessiven Approximation der Schwerpunkt auf den Zusammenhang zwischen Auflösung und Geschwindigkeit gelegt. Im Anhang A.2 ist das in MATLAB/Simulink realisierte Gesamtmodell eines 4-bit Wandlers beispielhaft angegeben. Dieses Modell kann erweitert werden (sinnvollerweise bis maximal 24 Bit). Durch die Untersuchung von unterschiedlichen Umsetzungsvarianten abhängig von den Anforderungen kann eine optimale Realisierungsstruktur gefunden werden.

In der Abbildung 4.8 sind auch weitere Komponenten zur Kommunikation von unterschiedlichen Daten und physikalischen Größen abgebildet. Die Signalanpassungsfunktionen dienen zur Darstellung der abgetasteten beziehungsweise ausgegebenen Signale in Form von geeigneten Werten. Die Umwandlung in eine auf die Maßeinheit Volt bezogene Größe dient zur Darstellung der abgetasteten Spannung (Ausgang des AD-Wandlers: binär kodierter Wert) als einem für die digitale Signalverarbeitung geeigneten Wert (hier Gleitkommadarstellung). Vor dem Modell des DA-Wandlers befindet sich die umgekehrte Umwandlung: Ein Ergebnis mit einem in der Signalverarbeitung erzeugten Zahlenformat (hier wieder Gleitkommadarstellung) wird in Form einer binär kodierten Zahlendarstellung an den verwendeten Wandler angepasst.

Die weiteren Modelle und deren Verfeinerungen beziehen sich auf die eigentlichen Funktionen der Signalverarbeitung. Im Regelungssystem wurden zwei Komponenten umgesetzt: eine Vorfilterung zur Verbesserung der Mess- und Regelgüte und die nachfolgende Regelung.

Im Rahmen dieser Arbeit wurden je ein klassischer FIR und IIR Tiefpassfilter modelliert. Mittels der Gleichungen 4.1 und 4.2 [Ja10] können die Algorithmen in Modelle umgesetzt werden. Außerdem entsteht die Möglichkeit, einige Standardfunktionen der digitalen Signalverarbeitung in Form von Bibliotheken in Entwicklungsumgebungen wie MATLAB/Simulink und LabVIEW zu benutzen. Dieses ermöglicht die genauere Analyse zur Auswahl eines für die konkrete Anwendung geeigneten Algorithmus und dessen Realisierungsstruktur.

FIR-Filter

$$y(k) = \sum_{i=0}^{B} b_i \cdot u(k-i) \qquad \textbf{(4.1)}$$

IIR-Filter
$$y(k) = \sum_{j=1}^{A} a_j \cdot y(k-j) + \sum_{i=0}^{B} b_i \cdot u(k-i) \qquad \textbf{(4.2)}$$

Eine weitere wichtige Aufgabe des technischen Systementwurfes ist die Untersuchung der von der Anwendung geforderten und optimalen Realisierungsstrukturen. Für die Dissertation wurden für die konkrete Anwendung unterschiedliche Regler-Konzepte basierend auf einem Polynomregler verwendet. Mittels der charakteristischen zeitdiskreten Übertragungsfunktion (Gl. 4.3 nach [RZ04]) kann deren Polynom im Modell umgesetzt werden. Durch dessen Parametrisierung (Koeffizienten a_i und b_j) konnten verschiedene Reglertypen abgebildet werden, z.B. PID-, Kompensations-, H∞- und IMC-Regler. Bei der Analyse und Auswahl eines geeigneten Reglers (z.B. bezüglich kurzer Einschwingzeit und Robustheit) stehen die festgelegten messtechnischen Anforderungen im Vordergrund. Die regelungstechnischen Vorgaben stellen meistens einen Zeitkonflikt dar, der einen Kompromiss bezüglich der Zeitkriterien benötigt.

$$G_R(z) = \frac{b_n + b_{n-1} \cdot z^{-1} + b_{n-2} \cdot z^{-2} + \cdots + b_0 \cdot z^{-n}}{a_m + a_{m-1} \cdot z^{-1} + a_{m-2} \cdot z^{-2} + \cdots + a_0 \cdot z^{-m}} \qquad \textbf{(4.3)}$$

Die modellbasierte Entwicklung bietet die Möglichkeit, eine Begrenzung des Lösungsraums zu finden. Im obengenannten Beispiel wurden deshalb nur Polynome von Reglern mit maximal 10. Ordnung betrachtet. Dadurch konnte für die untersuchte Funktion des Prototyps bezüglich der regelungstechnischen Lösungen eine einheitliche Realisierungsstruktur, nämlich für ein Polynom genau 10. Ordnung (eventuell nicht benötigte Koeffizienten werden auf Null gesetzt), verwendet werden (s. auch Kapitel 5).

4.3.1.2 Modelle von Sicherheitsfunktionen

Neben den oben dargestellten Modellierungen der Hauptkomponenten des Einbettenden und Eingebetteten Systems sind die im Abschnitt 4.2.3 eingeführten Sicherheitsfunktionen und deren Wichtigkeit in der Entwurfsdomäne Messtechnik in der vorliegenden Dissertation betrachtet worden. Beim technischen Systementwurf werden die im funktionalen Systementwurf festgelegten Sicherheitsfunktionen in Ergänzung zu den rein messtechnischen Funktionen verfeinert und im Zusammenhang mit anderen Systemkomponenten modellbasiert untersucht.

Die Algorithmen für die äußere Sicherheit sind entwurfsaufwendig (s. z.B. Übersicht der Advanced Encryption Standard- (AES-)Varianten [GCh01]) und sollen deshalb im Gesamtsystem unter Berücksichtigung des Ressourcenverbrauchs und des zeitlichen Aufwandes besonders analysiert werden. Darauf folgend wird eine Entscheidung bezüglich des für die konkrete Anwendung verwendeten Algorithmus getroffen. Diese muss im Kontext der gesetzlichen und weiteren Anforderungen (s. Abschnitt 2.3.2) gefällt werden.

In der Diplomarbeit von Bock [B12] wurden ausgewählte Sicherheitsfunktionen der AES-Verschlüsselung untersucht und modelliert. Diesem Standard liegt der Rijndael-Algorithmus zugrunde [DR14]. Abhängig vom Sicherheitsgrad kann eine Schlüssellänge von 128, 192 oder 256 Bit gewählt werden und mit Blockgrößen von 128, 192 oder 256 Bit kombiniert werden. In diesem Zusammenhang richtet sich die benötigte Iteration innerhalb des Algorithmus nach der Schlüssellänge. In [B12] wurde der Algorithmus mit einer Blockgröße und Schlüssellänge von je 128 Bit betrachtet, modelliert und getestet (s. Anhang A.2: Abb. A.5 und Abb. A.6).

Für diese Dissertation wurde, wiederum in der Diplomarbeit [B12], ein weiterer Algorithmus, der Twofish-Algorithmus [SKWW98], untersucht. Dieser wurde nicht in den AES-Standard übernommen, hat aber den kleinsten Ressourcenverbrauch und realisiert eine Sicherheitsarchitektur gegen noch unbekannte Angriffe. Er stellt einen 128 Bit-Blockverschlüssler mit akzeptierter Schlüssellänge bis zu 256 Bit dar. Als Modell wurde eine Klartextbox mit einem Schlüssel der Länge 128 Bit realisiert. Das Gesamtmodell besteht aus drei Teilen: Erzeugung eines Rundenschlüssels, Ver- und Entschlüsselung (s. Anhang A.2: Abb. A.7 und Abb. A.8).

Im technischen Entwurf soll neben der Untersuchung der Umsetzungsmöglichkeit auch die Wiederverwendung von verfügbaren Modellen beziehungsweise Funktionen berücksichtigt werden. Außer der nach ZEfIRA geforderten Wiederverwendung von Modellen aus dem Vorläufersystem ist an dieser Stelle die Verwendung von anderen verfügbaren Realisierungen gemeint. Da z.B. mehrere Umsetzungen des AES-Algorithmus in Form von wiederverwendbaren Bibliotheken, IP-Cores oder als C-Code existieren, ist es möglich eine Performance-Vergleichsanalyse durchzuführen, um eine optimale und für die konkrete Anwendung geeignete Realisierungsstruktur zu finden. In diesem Zusammenhang müssen zutreffende Anforderungen an die Metrologische Sicherheit erfüllt werden.

Der durchgeführte technische Systementwurf der weiteren im Abschnitt 4.2.3 genannten Sicherheitsfunktionen erwies sich als weniger komplex und aufwendig [MP14][Ra13]. Deswegen wird hier auf eine ausführliche Darstellung verzichtet (Konkreteres zur Realisierung enthält Abschnitt 5.2.4).

4.3.2 Plattformunabhängige Datentypanalyse

Bei der Realisierung und der Ausführung von Algorithmen der Signalverarbeitung auf einer Zielplattform ist es eine Herausforderung, mit reellen Zahlen zu operieren. Diese werden in der digitalen Technik mit einer bestimmten Genauigkeit dargestellt. In diesem Fall werden die sogenannten Maschinenzahlen zur Approximation von reellen Zahlen und elementaren arithmetischen Operationen verwendet. Für die Darstellung einer Zahl steht eine feste Anzahl von Bits zur Verfügung, dabei arbeitet man mit einem bestimmten Datentyp. [Zu06]

Beim technischen Systementwurf wird die Untersuchung von unterschiedlichen, für die Applikation notwendigen Datentypen durch das modellbasierte Prinzip möglich und unterstützt. In diesem Fall wird das Informationsverarbeitungssystem bei Berechnungen mittels verschiedener Datentypen als ganzes System beziehungsweise auch nur für Teilsysteme simuliert und die Ergebnisse werden validiert. Nach der Festlegung eines Datentyps ist die Abschätzung der Eignung für die zu realisierenden Algorithmen vor allem bezüglich der Genauigkeit und des Entwicklungsaufwandes auf hohem Abstraktionsniveau weitgehend plattformunabhängig möglich. Dabei kann es sinnvoll sein, in verschiedenen Teilsystemen mit unterschiedlichen Datentypen zu arbeiten.

Die Abbildung A.9 (im Anhang A.2) zeigt die unterschiedlichen Datentypen, die in der Praxis bei der Realisierung von Informationsverarbeitungsaufgaben verwendet werden. Im Folgenden werden die Repräsentationen und die Vor- und Nachteile jedes Datenformates dargestellt (s. auch [MB07]).

Der Datentyp „Integer“ ist Vertreter von ganzen Zahlen. Dieser wird häufig für die Darstellung von codierten Werten nach der analog-digitalen beziehungsweise digital-analogen Umwandlung verwendet.

Als Beispiel wird die analoge Spannung (U_M = 5,846 V) nach der 16-bit-AD-Umwandlung (Spannungsbereich ± 10 V entspricht Uref = 20 V) als 16-bit-Integer-Code (DigitAD = 19156) dargestellt. Die allgemeine Formel 4.4 zeigt die Berechnung der Integer-Darstellung abhängig von der Auflösung der verwendeten AD- bzw. DA-Umsetzern für die Spannungsumwandlung.

$$\mathbf{Digit_{AD/DA} = U_M \cdot \frac{2^n}{Uref}} \qquad \textbf{(4.4)}$$

mit $Digit_{AD/DA}$ – Darstellung einer reellen Zahl als Integer-Code; U_M – Messwert (Spannung als reelle Zahl), n – Auflösung des verwendeten AD- bzw. DA-Umsetzers.

Der Datentyp Integer ist bei Berechnungen mit wenigen elementaren mathematischen Operationen geeignet. Dabei ist die Implementierung aufwandsarm, benötigt wenige Ressourcen und hat kurze Laufzeiten. Bei der Realisierung von komplexen Algorithmen der Signalverarbeitung entstehen mehrere Nachteile. Zum einen ist nur die Darstellung eines kleinen Wertebereichs möglich. Zum anderen sind Algorithmen mit Rückkopplung (z.B. für Regelung, Steuerung, IIR-Filterung) nur in einfacher Form mit speziellen angepassten Zahlenbereichsverschiebungen möglich. Dabei entstehen bei jeder Operation Rundungsfehler (z.B. untersucht in [Hu12] anhand Gleitkomma-Arithmetik), die sich bei mehreren Operationen fortpflanzen (s. Abschnitt 4.3.3), den Gesamtfehler erhöhen und damit die Genauigkeit verringern.

Der Datentyp „Fixed-Point“ wird zur Darstellung von reellen Zahlen mit Festkomma verwendet. Es existieren unterschiedliche Möglichkeiten der Repräsentation von Festkommazahlen [MB07]. Dieser Datentyp wird häufig bei der Implementierung von komplexen Algorithmen der digitalen Informations- beziehungsweise Signalverarbeitung eingesetzt, da er leicht realisierbar ist, wenige Ressourcen braucht und gute Laufzeiten bei Berechnungen mit einer höheren Genauigkeit der Darstellung ermöglicht [Ag14]. Dabei ist eine Anpassung an die konkreten Algorithmen notwendig und das ist oft eine Herausforderung. Der Grund liegt darin, dass die Position des Kommas beziehungsweise die Länge der Nachkommastellen nach jeder mathematischen Operation durch geeignete Vorgehensweisen (Verschieben beziehungsweise Abschneiden von Bitpositionen) festgelegt werden muss. Das Ergebnis kann mehr binäre Stellen haben als die Eingangsoperanden. Dabei kann eine falsche Festlegung des Kommas zu Informationsverlusten, die eventuell gravierend sind, führen. Außerdem braucht man in komplexen Filter-, Regelungs- und Steuerungsalgorithmen der digitalen Signalverarbeitung manchmal die Darstellung und die Berechnung von sehr kleinen und/oder sehr großen Zahlen (z.B. 10^{-30}, 10^{30}). Diese sind mit der Festkommadarstellung nicht möglich, wenn man sich im Standard-Wertebereich dieser Darstellung bewegt.

Der Datentyp „Floating-Point“ ermöglicht ebenfalls die Datendarstellung mit einer festen Anzahl von Stellen, das Komma wird aber normalisiert verschoben. Diese Darstellung entspricht einer modifizierten Exponentialschreibweise wie in der Mathematik. Die tatsächliche Repräsentation einer reellen Zahl besteht dabei aus einem Vorzeichen-Bit, den Mantissen-Bits und den Exponenten-Bits (entsprechend der Abb. A.10 im Anhang A.2). In den meisten Rechnersystemen wird das IEEE Format (Institute of Electrical and Electronics Engineers) als eine genormte Gleitkommadarstellung (auch Fließkommadarstellung bekannt) verwendet [IE14].

Es gibt die Darstellung mit einer einfachen (Single) und einer doppelten (Double) Genauigkeit. Außerdem erlaubt die IEEE-Gleitkommadarstellung spezielle Werte: *Inf* für unendlich, z.B. wenn eine Zahl durch Null dividiert wird; *NaN* für „not-a-number“, wenn eine Operation undefiniert oder unbestimmt ist [MB07].

Die Verwendung von Gleitkommazahlen ermöglicht die Darstellung eines erheblich größeren Zahlenbereiches als mit der Festkommadarstellung. Allerdings kann die Genauigkeit bei der gleichen Bitlänge kleiner als bei der Festkommadarstellung sein, wegen der Integration des Exponenten in die Bitdarstellung auf Kosten der Länge der Mantisse.

Wegen der Flexibilität bei den Berechnungen wird die Gleitkommadarstellung immer häufiger in komplexen Informationsverarbeitungssystemen verwendet [MKGP13]. Sie verringert dabei den Entwicklungsaufwand, weil bei den meisten Algorithmen die Veränderung der Genauigkeit durch die Operationen und die Ausnutzung des Wertebereichs nicht betrachtet werden muss.

Im Anschnitt 4.4.1 wird dieser Datentyp bezüglich der FPGA-Realisierungsplattform detaillierter betrachtet, da er auf diesem Niveau Untersuchungen unter dem Gesichtspunkt der dort zu berücksichtigenden Restriktionen verlangt.

Das folgende Beispiel zeigt, welchen Einfluss die verwendeten Datentypen auf die Abweichungen der Berechnungsergebnisse haben. Verwendet wird ein in LabVIEW modelliertes und simuliertes Regelungssystem mit unterschiedlichen Datentypen. Die Abbildung 4.9 zeigt die resultierenden errechneten Spannungswerte (Zeitverläufe der Stellgröße und Positionsspannung) des Regelungssystems (Darstellung eines PID-Reglers mit einem Polynom 10. Ordnung). Hierzu sind die Verläufe der ersten 0,15 s (beim Verlauf der Stellgröße) und der ersten 0,35 s (beim Verlauf der Positionsspannung) nach einer sprungförmigen Belastung für die drei Datentypen Single, Double Fixed-Point dargestellt.

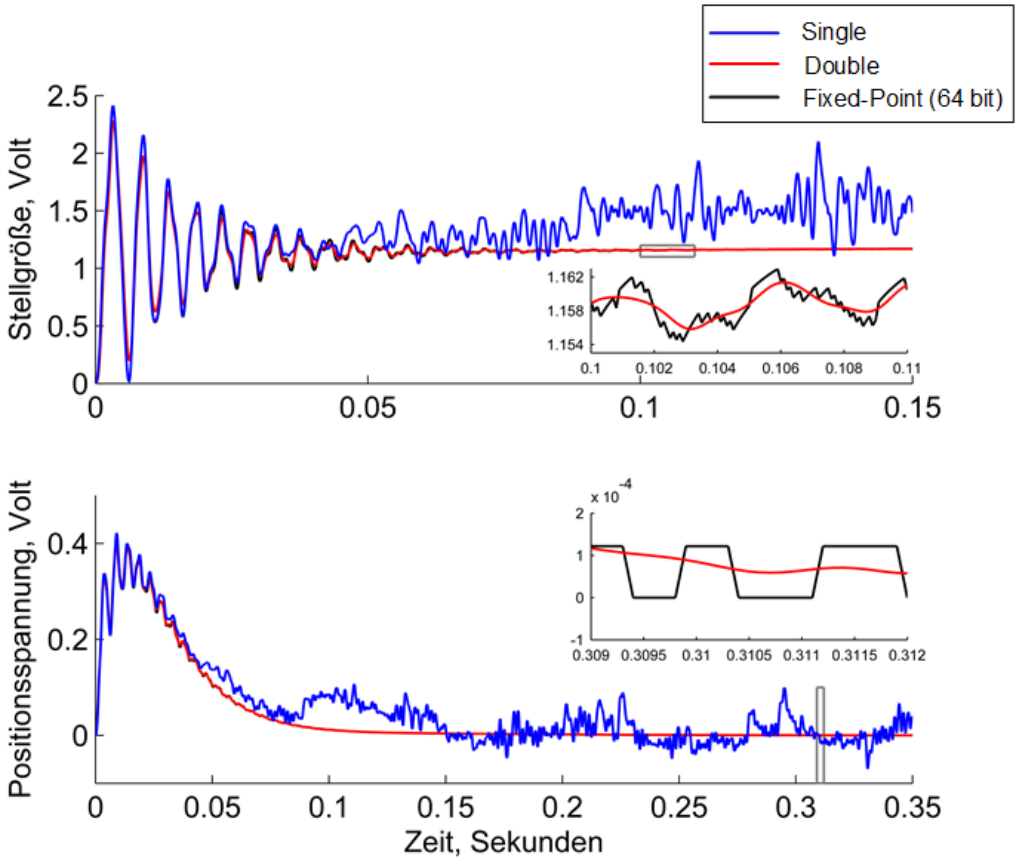


Abbildung 4.9: Unterschiedliche Zeitverläufe des simulierten Regelungssystems

Die Ergebnisse zeigen, dass die Verwendung von Floating-Point-Zahlen mit doppelter Genauigkeit und von Fixed-Point-Zahlen beide die gestellten Anforderungen bezogen auf die Messunsicherheit gewährleisten (s. genauer im Kapitel 5). Beim ersten Datentyp erfolgt eine geringfügige Verbesserung beim Verlauf der Stellgröße und bei der bleibenden Regelabweichung (s. angezeigte Bereiche in Abb. 4.9). Außerdem konnten nur mit dem

Datentyp „Double“ bei der Parametrisierung des Reglers sehr kleine beziehungsweise sehr große Zahlen (z.B. Koeffizienten des Reglers: 10^{32}) verwendet werden. Die Umsetzung eines Regelungsalgorithmus mittels Festkommazahlen erforderte hier die genaue Analyse und Anpassung bezüglich der möglichen Überläufe. Das bedeutet, dass nach jeder ausgeführten mathematischen Operation die Veränderung des Wertebereiches auf Grund des möglichen Maximums des Ergebnisses bestimmt werden muss. Diese Ermittlung ist für jeden Parametersatz durchzuführen. Das beschränkt insbesondere die Flexibilität bei der Untersuchung verschiedener Algorithmen und Parametersätze ein. Weiterhin ist eine genauere Kenntnis des internen Aufbaus des Datentyps notwendig. In diesem Beispiel mussten die Ergebnisse von 20 Multiplikationen und 10 Additionen angepasst werden.

Die Realisierung durch Floating-Point-Zahlen mit einfacher Genauigkeit (Single) erreicht die für die geforderte Messunsicherheit notwendige Auflösung nicht.

4.3.3 Fehleranalyse

Die Entwicklung von Informationsverarbeitungssystemen besonders für die messtechnische Anwendung benötigt eine Möglichkeit zur mathematischen Abschätzung von systematischen Fehlern in der Messkette. Bei der modellbasierten Entwicklung kann die Behandlung von unterschiedlichen Fehlerquellen unterstützt werden. Das Ziel dieses Abschnittes ist es, zu zeigen, wie die Methode des Fehlerfortpflanzungsgesetzes bei dem technischen Systementwurf berücksichtigt und integriert werden kann. Im Prozess ZEfIRA ist die Realisierung von Funktionen der Fehleranalyse ein Aspekt zum Nachweis der Metrologischen Sicherheit dieses Entwurfsschrittes.

Es existiert eine Reihe von Gründen, die zu Fehlerquellen führen. Bei den Berechnungen entstehen Fehler des mathematischen Modells, nicht vermeidbare Fehler der Lösung durch die Ungenauigkeit von Eingangsgrößen, Fehler in den verwendeten Algorithmen und Prinzip-basierte Rechenfehler der Informationsverarbeitung. Es ist deshalb die Entwicklung von Modulen beziehungsweise speziellen Komponenten notwendig, die eine Fehleranalyse parallel zu den benötigten Berechnungen ermöglichen. In der Dissertation wurden beispielhaft Module für die modellbasierte Fehleranalyse eines Analog-Digital-Umsetzers und für die numerische Analyse von modellierten Algorithmen der Signalverarbeitung umgesetzt. Diese werden in den folgenden Teilabschnitten genauer beschrieben.

4.3.3.1 Modul zur Berechnung der Genauigkeit eines AD-Wandlers

Die erste Komponente des Eingebetteten Systems in der Messkette ist ein Analog-Digital-Wandler. Wie oben beschrieben wurde, erfordert die Verwendung eines konkreten Wandlers eine Untersuchung bezüglich der Anwendbarkeit und der Fehleranalyse, da dieser die

Messunsicherheit des Ergebnisses beeinflusst. Die modellbasierte Berechnung des gesamten Fehlers, den die analog-digitale Umwandlung bei der Fehlerfortpflanzung einbringt, ist schwierig. Die Einflussfaktoren eines AD-Wandlers sind vor allem abhängig vom Eingangssignal und dessen Frequenz sowie der Abtastrate, Temperatur und Feuchte, so dass diese Parameter zu berücksichtigen sind. Die Fehlerquellen sind unterschiedlicher Natur, und ihnen liegen verschiedene physikalische Gesetze zugrunde. Deren gemeinsame Betrachtung und Behandlung ist zumeist kompliziert [JCGM10]. In vielen Fällen wird für die praktische Anwendung der in der Dokumentation vom Hersteller angegebene maximale Fehler des Bauelementes benutzt. Für eine bessere Bestimmung der Genauigkeit wird oft auch zusätzlich eine weitere Methode genutzt. Dabei werden die für die unterschiedlichen Fehlermöglichkeiten zu berücksichtigenden Werte durch praktische Messungen unter den Bedingungen des beabsichtigen Anwendungsfalls experimentell ermittelt.

Zur modellbasierten Abschätzung der Messunsicherheit von AD-Wandlern gibt es eine Reihe von Arbeiten, z.B. die Dissertation von Vallant „Modellbasierte Entzerrung von Analog/Digital-Wandler-Systemen“ [Val14]. Durch die Modellierung von Fehlermechanismen an der AD-Schnittstelle wurden hier Korrekturmodelle erzeugt, die effektiv zur Dynamikbereichserweiterung digitaler Empfänger verwendet werden können.

Für diese Arbeit wurde das prinzipielle Vorgehen für eine modellbasierte Untersuchung (hier in MATLAB/Simulink) exemplarisch untersucht. Das im Forschungsprojekt von Ravi Mambally D. [Mam14] umgesetzte Modul ermöglicht die Berechnung der oberen Schranke des absoluten Fehlers für verschiedene Wandler (s. Abb. 4.10). Dabei lag der Schwerpunkt weniger bei der algorithmischen Feinheit sondern mehr beim modellbasierten Vorgehen zur Integration in den Entwurfsprozess ZEfIRA.

Bei der Berechnung des Fehlers werden neben dem mit dem Wandler direkt verbundenen Quantisierungsfehler die folgenden Fehlergrößen aus der Dokumentation des verwendeten AD-Wandlers berücksichtigt:

- Offsetfehler (Nullpunktfehler) – Abweichung der tatsächlichen von der idealen Kennlinie im Nullpunkt der analogen Größe (in Volt);
- Verstärkungsfehler (engl. *Gain Error*) – Anstiegsdifferenz zwischen idealer und realer Übertragungsfunktion ohne Offset-Fehler (in Volt);
- Integraler Linearitätsfehler (INL) – Abweichung der realen Kennlinie von der idealen (in LSB);
- Differentielle Nichtlinearität (DNL) – gibt an, um welchen Betrag die Breite der einzelnen Quantisierungsstreifen vom Sollwert abweicht (in LSB).

Das aus diesen ermittelte Ergebnis (s. Abb. 4.10: *Total Standard Uncertainty in Volts*) kann bei der Fehlerfortpflanzungsanalyse weiter verwendet werden.

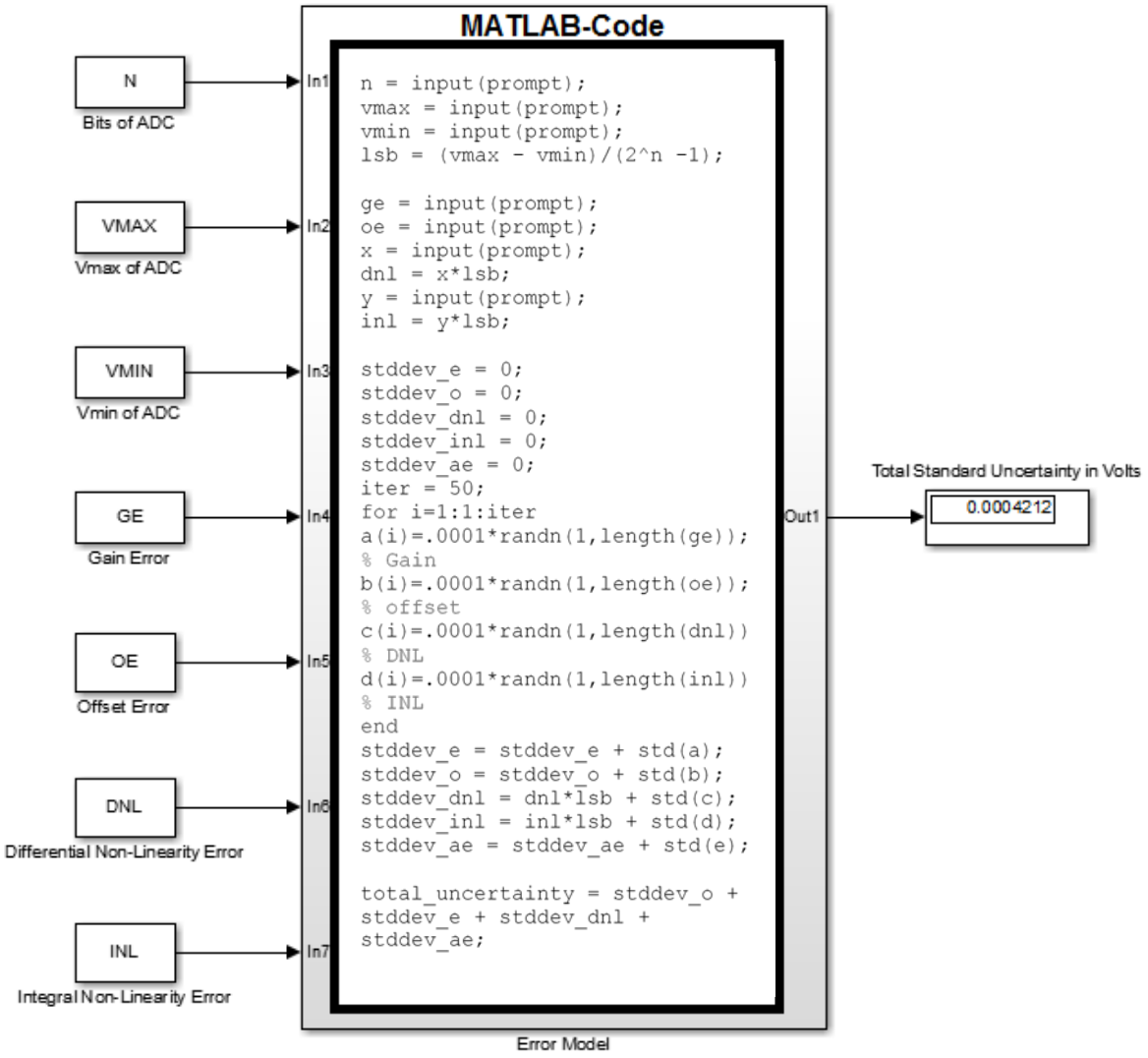


Abbildung 4.10: Modul zur Berechnung des Fehlers eines AD-Wandlers

Das realisierte Programm (MATLAB-Code) zur Berechnung des Fehlers eines AD-Wandlers kann sowohl in MATLAB/Simulink verwendet oder in einem anderen Tool, z.B. LabVIEW, implementiert werden.

4.3.3.2 Fehlerfortpflanzung durch mathematische Operationen

Die Realisierung von unterschiedlichen Algorithmen der Signalverarbeitung soll unter der Berücksichtigung von Prinzip-basierten Rechenfehlern der Informationsverarbeitung durchgeführt werden. Dafür wurden in der Etappe des technischen Systementwurfes spezielle Komponenten entwickelt, die eine numerische Analyse der verwendeten mathematischen Operationen ermöglichen. Entsprechend der Tabelle 4.1 wurden die Grundoperationen mit ihren Unsicherheiten umgesetzt. Angenommen, es werden die Werte ***a*** und ***b*** mit entsprechenden Unsicherheiten ***u(a)*** und ***u(b)*** gegeben.

Das Ergebnis ***c*** hat in diesem Fall die geschätzte Unsicherheit ***u(c)***.

Mathematische Operation	***Gleichung***	***Berechnung der Unsicherheit des Ergebnisses***
Addition	$c \pm u(c) = \big(a \pm u(a)\big) + (b \pm u(b))$	$u(c) = \sqrt{(u(a))^2 + (u(b))^2}$
Subtraktion	$c \pm u(c) = \big(a \pm u(a)\big) - (b \pm u(b))$	$u(c) = \sqrt{(u(a))^2 + (u(b))^2}$
Multiplikation	$c \pm u(c) = \big(a \pm u(a)\big) \cdot (b \pm u(b))$	$\frac{u(c)}{c} = \sqrt{(\frac{u(a)}{a})^2 + (\frac{u(b)}{b})^2}$
Division	$c \pm u(c) = \frac{a \pm u(a)}{b \pm u(b)}$	$\frac{u(c)}{c} = \sqrt{(\frac{u(a)}{a})^2 + (\frac{u(b)}{b})^2}$
Potenz	$c \pm u(c) = \big(a^n \pm u(a^n)\big)$	$\frac{u(c)}{c} = \frac{u(a^n)}{a^n} = \frac{n \cdot u(a)}{a}$
Wurzel	$c \pm u(c) = \sqrt[n]{a} \pm u(\sqrt[n]{a})$	$\frac{u(c)}{c} = \frac{u(\sqrt[n]{a})}{\sqrt[n]{x}} = \frac{u(a)}{n \cdot a}$

Tabelle 4.1: Fehlerfortpflanzung von mathematischen Operationen (nach [Si04])

Die realisierten Komponenten werden bei der Modellierung und Untersuchung von Algorithmen anstelle von nicht fehlerbehafteten exakten mathematischen Operationen verwendet. Der Gesamtfehler entsteht dabei durch die Summierung aller betrachteten Fehlerquellen. Für diese Arbeit wurde das prinzipielle Vorgehen (wie auch schon im Abschnitt 4.3.3.1) für eine modellbasierte Untersuchung, hier in LabVIEW, exemplarisch untersucht [Ram14]. Dabei lag der Schwerpunkt auch hier weniger bei der algorithmischen Feinheit sondern mehr beim modellbasierten Vorgehen zur Integration in den Entwurfsprozess ZEfIRA. In diesem Zusammenhang wurden Bibliotheken erstellt, die in nachfolgenden prototypischen Entwicklungen beim technischen Systementwurf wiederverwendet werden können (s. Abschnitt 4.3.5).

Es gibt eine Reihe von Literaturquellen, die sich mit detaillierteren Betrachtungen der Fehlerfortpflanzung durch mathematische Operationen auseinandersetzen [Bo99]. Zur Realisierung von komplexen Algorithmen mit Gleitkomma-Arithmetik kann man z.B. in [BCV06] Ansätze für die in Bezug auf Genauigkeit, Hardware-Ressourcen (FPGA-Realisierung) und Ausführungszeit effizienten und optimierten Gleitkomma-Implementierungen finden. Als Weiteres stellt die Dissertation von Huynh [Hu12] ein entwickeltes MATLAB-basiertes Framework für die Rundungsfehleranalyse sowie für die Auswertung des numerischen Bereiches von beliebigen Gleitkomma-Algorithmen mittels eines Fehlermodells, das auf der Affine Arithmetik basiert, dar.

4.3.4 Zeitanalyse

Innerhalb eines Informationsverarbeitungssystems in der betrachteten Anwendungsdomäne sind dessen Funktionen mit den Anforderungen der Prozesssteuerung eng verbunden. Hierzu ist die Einhaltung von Zeitschranken (Deadlines) unbedingt erforderlich (harte Echtzeitbedingungen [BST10]). Daraus ergeben sich die zeitlichen Bedingungen, deren Einhaltung im technischen Systementwurf auf Grund von Verfeinerungen analysiert werden können. Die Überprüfung von Bedingungen ist hier noch nicht vollständig möglich, sondern erst am Ende des plattformspezifischen Entwurfs.

Es gibt mehrere Möglichkeiten, um ein gefordertes Verhalten einer Komponente beziehungsweise eines Teilsystems zu beschreiben. In diesem Zusammenhang werden quantitative und qualitative Zeitbedingungen berücksichtigt, die als eine Aussage über die Korrektheit des Systementwurfs dienen. Die qualitative Zeitbedingung beschreibt ein gefordertes Verhalten mit der Angabe von Zeitwerten, im Gegensatz dazu werden bei den qualitativen Zeitbedingungen die geforderten Ablaufreihenfolgen ohne Angabe eines Zeitwertes festgelegt. Außerdem wird für einige Algorithmen der Signalverarbeitung (z.B. Regelung) die zeitliche Bedingung in Form einer Tastperiode angegeben. Die Ausführungszeit kann in vielen Fällen realitätsnäher durch eine untere und eine obere Zeitdauergrenze beschrieben werden. Das berücksichtigt, dass der exakte Zeitwert nicht bekannt oder variabel sein kann. Das zeitliche Verhalten wird dabei mit Hilfe von Zeitdauerintervallen modelliert. Da innerhalb eines technischen Prozesses einige Vorgänge sequenziell und andere parallel ablaufen, gelten Grundregeln für beide Fälle (s. Abb. 4.11).

Komplexere Systeme werden dann auf diese zurückgeführt.

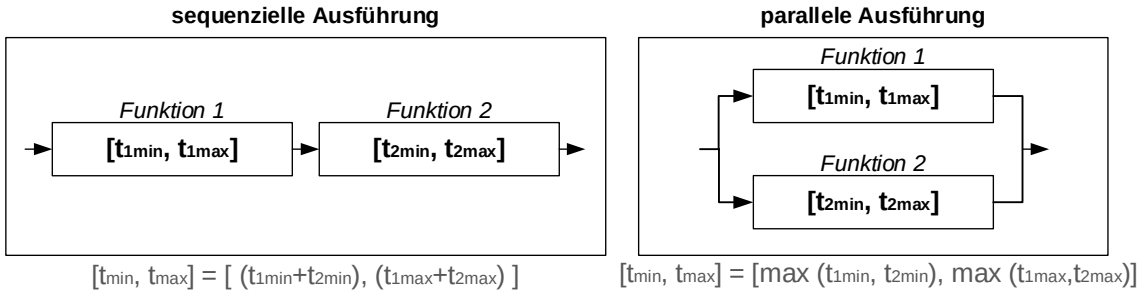


Abbildung 4.11: Zeitbestimmung bei sequentieller und paralleler Ausführung

Für die Zeitanalyse können beim technischen Systementwurf unterschiedliche Zeitmodelle eingesetzt werden. Diese müssen das Zeitverhalten geeignet repräsentieren und analysieren. Außerdem können durch die Verwendung von formalen Beschreibungsmitteln und Methoden Verhaltenseigenschaften verifiziert werden [Pa09]. Es existiert eine Reihe von Arbeiten, die sich auf die zeitliche Analyse in komplexen Systemen beziehen. Z.B. wurde in der Dissertation von Licht [Li04] ein Verfahren zur zeitlichen Analyse von

UML-Modellen beim Entwurf von Automatisierungssystemen beschrieben. Dabei wurde die Anwendung von Realzeitautomaten neben den verbreiteten zeitbehafteten Petri-Netzen [DH01] und der zeitbehafteten Prozessalgebra vorgeschlagen. An einem Beispiel konnten das zeitlichen Verhalten und die Zeitbedingungen in einem UML-Modell dargestellt werden.

Die Etappe des technischen Systementwurfes mit der durchgeführten Zeitanalyse dient als eine Vorlage im verfeinerten Entwurf. Durch die plattformabhängige Untersuchung kann man eine genauere Zeitbewertung durchführen. In diesem Zusammenhang wird in der Dissertation die formale Methode der gefärbten Zeitintervall-Petri-Netze benutzt (s. Abschnitt 4.4.3).

4.3.5 Realisierung von wiederverwendbaren Modellen

Eine wichtige Forderung nach dem ZEfIRA-Prozess ist, wiederverwendbare Modelle zu entwickeln und zu realisieren. Diese können für die weiteren Generationen von Prototypen und Produkten eine wichtige Rolle spielen. Der vorliegende Abschnitt dient zur Darstellung von entworfenen Modellen der Komponenten des messtechnischen Systems, die im Laufe der Arbeiten für die Dissertation in Form von wiederverwendbaren Bibliotheken umgesetzt wurden.

Im Weiteren werden als Beispiel die Umsetzungen in der Entwicklungsumgebung LabVIEW unter vorwiegender Nutzung der dort vorhandenen Komponenten dargestellt. Die LabVIEW-Modelle wurden bei der Entwicklung des applikationsspezifischen Prototyps benötigt (s. Kapitel 5). Erstellung von wiederverwendbaren Modellen in Form von Bibliotheken kann aber auch in MATLAB/Simulink und unter Verwendung von Programmiersprachen (z.B. einige Funktionen als C-Code) alternativ oder in Kombination durchgeführt werden.

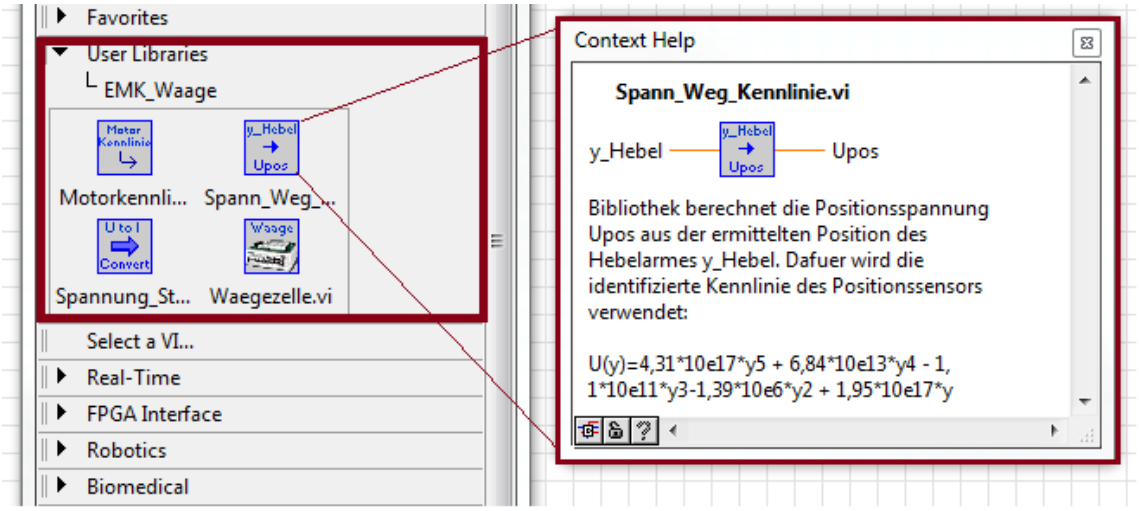


Abbildung 4.12: Ausschnitt aus den Bibliotheken des Einbettenden Systems

Die erste Gruppe von Bibliotheken (s. Abb. 4.12) betrifft die Komponenten des Einbettenden Systems. Die Umsetzung dieser Module erfolgt auf Basis von Informationen der Systemidentifizierung und durch die Modellierung und Beschreibung der Bewegungsdynamik der Wägezelle als Zwei-Balken-System (s. Abschnitt 4.3.1.1). Jede Bibliothek ist geeignet dokumentiert (Abb. 4.12 am Beispiel: Spannung-Weg-Kennlinie) und kann für andere Wägesysteme angepasst werden.

Die zweite Gruppe ist in der Abbildung 4.13 zu sehen. Sie enthält die modellierten Schnittstellen für die Kommunikation zwischen den Einbettenden und Eingebetteten Systemen. Neben den Modellen der Analog-Digital- und Digital-Analog-Wandler sind die Signalanpassungsfunktionen und das Modul zur Berechnung des Fehlers des verwendeten AD-Wandlers zu sehen. Letzteres ist in Abbildung 4.13 mit entsprechender Beschreibung präsentiert.

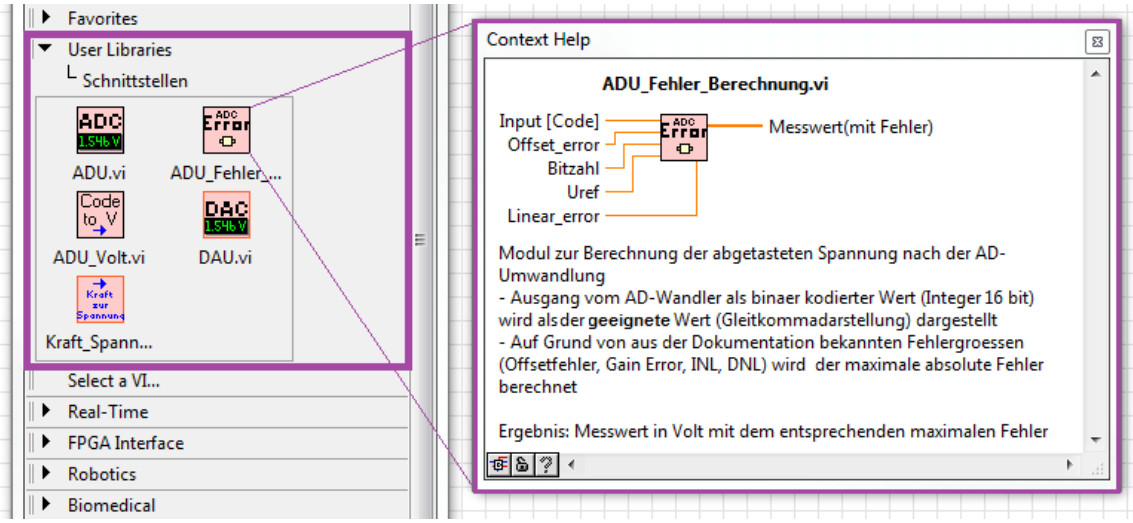


Abbildung 4.13: Bibliotheken von Schnittstellen

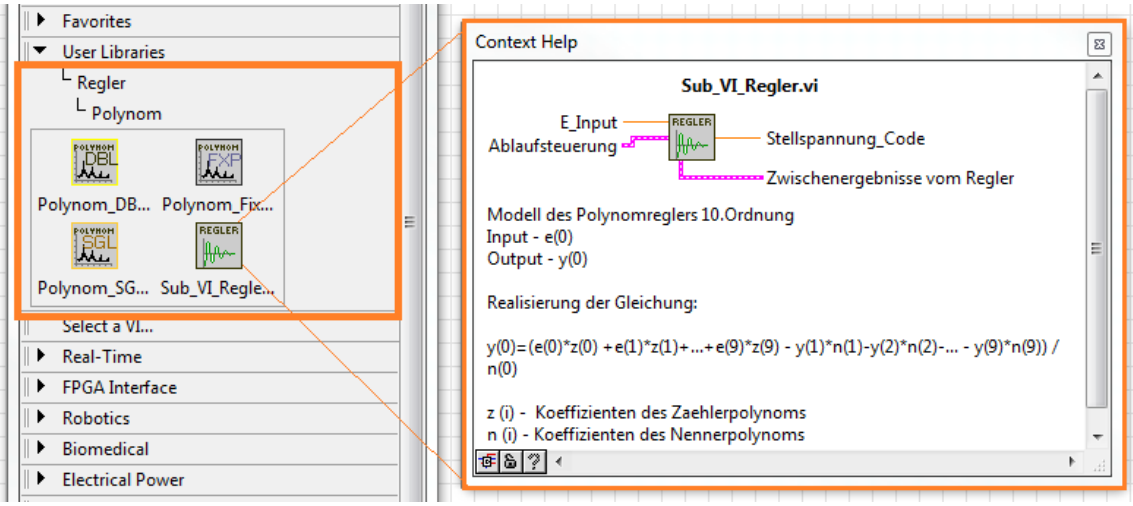


Abbildung 4.14: Bibliotheken der Signalverarbeitung (Regler)

Die Abbildung 4.14 zeigt eine Komponente der Signalverarbeitung, den Regelalgorithmus. Die Umsetzung erfolgte durch die Modellierung des Polynoms (nach der Gl. 4.3) mittels LabVIEW-Komponenten. In dieser Bibliothek stehen unterschiedliche Berechnungsvarianten abhängig von den Datentypen (Fest- oder Fließkomma-Arithmetik) zur Verfügung.

Die folgende Gruppe von Bibliotheken (Abbildung 4.15) wurde entsprechend dem Abschnitt 4.3.3.2 (Operationen mit zugehöriger Fehlerfortpflanzung) entwickelt. Es werden die Komponenten zur numerischen Analyse von verschiedenen mathematischen Operationen abhängig von den verwendeten Datentypen gezeigt. Als Beispiel stellt die Abbildung 4.15 die auf Basis der Tabelle 4.1 realisierten Operationen der Fließkommaarithmetik (Double) dar. Diese werden für die Fehleranalyse bei Modellierungen von Algorithmen der Signalverarbeitung statt der typischen mathematischen Operationen verwendet.

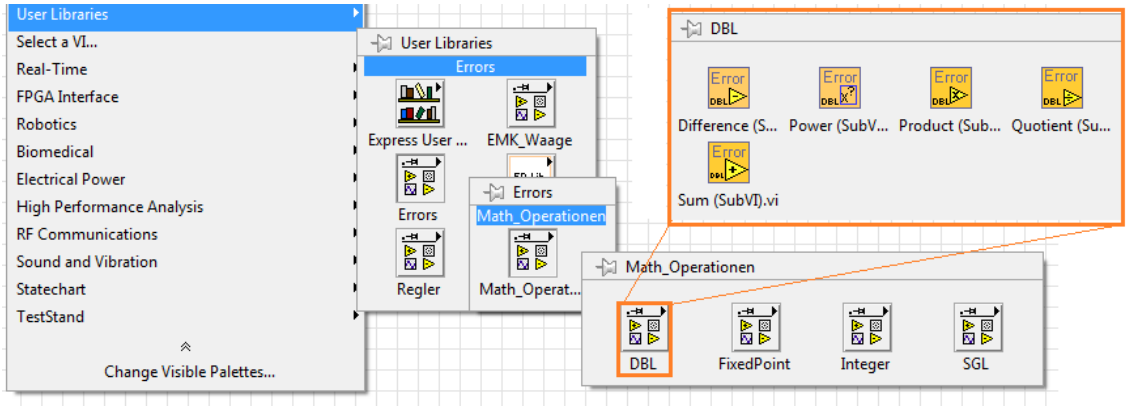


Abbildung 4.15: Bibliotheken der Fehleranalyse von mathematischen Operationen

4.4 Verfeinerter technischer Entwurf für rekonfigurierbare Hardware

Nach der Partitionierung in Einbettendes und Eingebettetes System, die im technischen Entwurf durchgeführt wird, beschäftigt sich diese Dissertation mit dem verfeinerten technischen Entwurf des Eingebetteten Systems. Dessen nach den entwickelten Prinzipien in den oberen Ebenen erarbeitete, beschriebene und modellierte Komponenten werden in dieser Entwurfsetappe plattformabhängig verfeinert und verifiziert. Der Schwerpunkt der Arbeit bei Realisierungsplattformen liegt in der Verwendung von hochleistungsfähiger und rekonfigurierbarer Hardware in Form von FPGAs. Die Aufgabe des modellbasierten Verfeinerungsprozesses ist die Abbildung von Modulen beziehungsweise Systemteilen auf plattformspezifische Modellelemente. Diese sollen die mit ihnen assoziierten Artefakte und Mechanismen für eine automatisierte Modell-zu-Code-Transformation ermöglichen.

In der Dissertation von Müller wurden Methoden und Prinzipien der plattformspezifischen Modellierung für verteilte rekonfigurierbare Systeme dargestellt [Mü12]. Es wurde die Beziehung der Datenflussdefinition und der Komponentendefinition zur Abbildung von plattformunabhängigen Spezifikations- auf FPGA-spezifische Systemmodelle erarbeitet. Zur Erstellung eines plattformspezifischen Modells einer Funktionskomponente werden Elemente des Datenpfades (*Data path model*) und des Steuerpfades (*Control path model*) zusammengesetzt. Diese sind Modelle von spezifizierten Algorithmen (Kernfunktionen), Pufferstrukturen für die Argumente und Ergebnisse, Statussignalstrukturen und Datengültigkeitssignale. Die letzteren spielen eine besondere Rolle in der plattformspezifischen Modellierung, da sie die Interaktion der Komponente sicherstellen und das interne Verhalten beim realen Ablauf steuern. Das in der Dissertation von Müller dargestellte Konzept konnte exemplarisch durch in MATLAB/Simulink realisierte Tool-Implementierungen an einem Beispiel aus der mess- und regelungstechnischen Anwendungsdomäne gezeigt werden.

Basierend auf dieser Dissertation und eigenen Untersuchungen stellt der vorliegende Abschnitt die Besonderheiten bezüglich der Realisierung/Implementierung von für die Applikation spezifizierten Algorithmen auf FPGAs im Prozess ZEfIRA dar.

Die Abbildung 4.16 zeigt die betrachteten Etappen des verfeinerten technischen Entwurfs im W-Prozess. Bei der Verfeinerung des im technischen Grobentwurf spezifizierten Eingebetteten Systems werden mehrere iterative Prozesse benötigt.

Zu Einem wird die Realisierbarkeit überprüft, also ob die gewünschten Algorithmen der Signalverarbeitung auf einer im Entwurfsprozess ausgewählten FPGA-Plattform umgesetzt werden können. Dafür sind Datentyp-, Ressourcen- und Zeitanalysen durchzuführen. In der Regel entsteht bei den Optimierungsaufgaben eine Konfrontation zwischen Berechnungs- und Leistungseigenschaften und den zur Verfügung stehenden Ressourcen. Als Ergebnis wird eine Systempartitionierung in Software-, Hardware- und FPGA-Module getroffen.

Die Feststellung, dass eine Realisierbarkeit mit dem ausgewählten FPGA nicht möglich ist, kann zu erneuter Auswahl, zur Verwendung mehrerer FPGAs oder auch zu Anpassungen beziehungsweise Änderungen im groben technischen Entwurf, im funktionalen Entwurf sowie zu anderer Partitionierung zwischen dem Einbettenden und Eingebetteten System und zu Änderungen in den Anforderungen führen (s. Abb. 4.16: **2** und **5**).

Ein weiterer Aspekt der iterativen Verfeinerung ist von Entscheidungen der Partitionierung des Eingebetteten Systems abhängig. Die spezifizierte Applikationsfunktionalität wird diesen entsprechend unterschiedlich modelliert und verschiedenen Implementierungskomponenten zugeordnet. Auch hier ist es möglich, dass eine Änderung bzw. Anpassung in den oberen Ebenen durchgeführt werden muss.

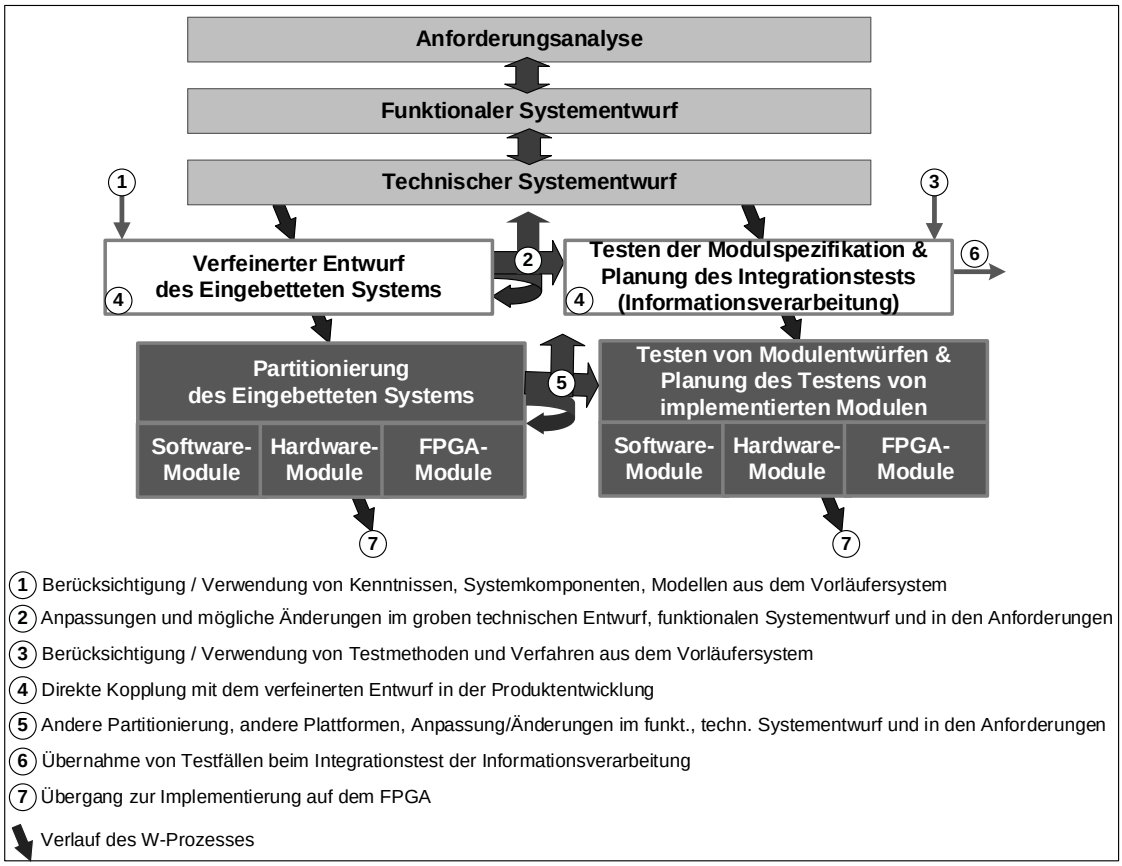


Abbildung 4.16: Verfeinerter technischer Entwurf des Eingebetteten Systems

Im verfeinerten technischen Entwurf muss, genau wie auf den anderen Ebenen des Entwicklungsprozesses, die Wiederverwendung von Kenntnissen beziehungsweise Modellen aus dem Vorläufersystem unterstützt werden (s. Abb. 4.16: **1** und **3**). Hierbei erhöht sich der Wiederverwendungsgrad durch die Integration von allgemeinen FPGA-basierten Lösungen (z.B. IP-Cores von FPGA-Herstellern). Dieses ermöglicht eine effektive Untersuchung und Umsetzung der gewünschten Funktionen. Die plattformabhängige modellbasierte Entwicklung ermöglicht die Abschätzung der Eignung der spezifizierten und im technischen Systementwurf umgesetzten Algorithmen bezüglich des Ressourcen- und Zeitverbrauchs sowie des Realisierungsaufwandes auf einem oder eventuell mehreren FPGAs. Diese Eigenschaften sind notwendigerweise im Zusammenhang zu betrachten. Deshalb werden beim verfeinerten Entwurf Ressourcen- und Zeitanalyse in Abhängigkeit zueinander und abhängig von den verwendeten Datentypen durchgeführt.

Die modernen Entwicklungsumgebungen (z.B. MATLAB/Simulink und LabVIEW) unterstützen verschiedene Möglichkeiten zur Verifikation und Validierung der beim technischen Entwurf realisierten Spezifikation. Durch die integrierten Simulatoren ist es meistens möglich, das funktionale Systemverhalten mit simulierten und realen Ein- und Ausgängen zu überprüfen. Die Analyse des exakten Zeitverhaltens, ebenso wie die des Ressourcenverbrauchs ist aber sehr von der verwendeten FPGA-Plattform abhängig.

4.4.1 Plattformspezifische Datentypaspekte

4.4.1.1 Eignung von Datentypen bei der FPGA-Realisierung

Im Abschnitt 4.3.2 wurden unterschiedliche Datentypen dargestellt, die zur Realisierung von Signalverarbeitungsaufgaben in der Praxis verwendet werden. An dieser Stelle werden die Vor- und Nachteile jedes Typs für ein FPGA-basiertes Eingebettetes System betrachtet.

Die Berechnung von Algorithmen mit wenigen elementaren mathematischen Operationen, aber auch festgelegten Teilfunktionen und bekannten Wertebereichen für die Parameter, ist aus vielen Gründen mit den Datentypen Integer und Fixed-Point sinnvoll. Zum einen werden diese beiden Datenformate für alle FPGA-Plattformen unterstützt und sind dadurch leicht anwendbar. Als Weiteres ermöglichen sie schnelle Berechnungen von gewünschten Funktionen und verbrauchen dabei wenige FPGA-Ressourcen.

Da der Datentyp Integer einen kleineren Wertebereich besitzt und zur Darstellung von Werten mit hoher Auflösung nicht geeignet ist, wird empfohlen, diesen nur in Berechnungen mit ganzen Zahlen oder in festgelegten Funktionen (z.B. zur Darstellung von codierten Werten nach einer AD-Umwandlung) zu verwenden.

Im Fall der Realisierung von komplexeren Algorithmen der Informationsverarbeitung werden diese häufig mit dem Festkomma-Datentyp umgesetzt. Diese Algorithmen können die Verwendung von sehr vielen Rechenoperationen pro Abtastzeit benötigen. Es gibt dabei Nachteile durch die Festkomma-Arithmetik, wenn es um die Realisierung von flexiblen und komplexen Funktionen bei der Berechnung mit hoher Auflösung geht. In vielen Aufgaben der Regelungstechnik werden Algorithmen zur Beobachtung und Schätzung einer oder mehrerer Zustandsgrößen benötigt (z.B. Kalman-Filter als Beobachter im Regelungssystem der NPMM [Am10]). In diesem Fall sind die Zustandswerte des Algorithmus variabel und können als sehr große beziehungsweise sehr kleine Zahlen auftreten, deren benötigter Wertebereich meistens nicht abschätzbar ist. Daraus folgend wird die Verwendung des Datentyps Fixed-Point nur bei der Realisierung von Algorithmen mit bekannten und/oder abschätzbaren Datenbereichen empfohlen. Hierzu werden aber geeignete Anpassungen der Wertebereiche nach jeder mathematischen Operation durchgeführt.

Eine flexible Alternative bietet die Verwendung der Fließkomma-Arithmetik. Außer den genannten Eigenschaften (s. Abschnitt 4.3.2), ist es wichtig zu betrachten, dass dieser Datentyp erheblich mehr Realisierungsaufwand und Ressourcen als andere Datentypen benötigt. Die FPGA-basierten Plattformen unterstützen bisher keine vorbereiteten Funktionsblöcke zur Berechnung in Gleitkommadarstellung einfacher oder doppelter Genauigkeit. Die Realisierung von Algorithmen der Signalverarbeitung wird dadurch erschwert,

dass die Verwendung von Fließkomma-Arithmetik auf dem FPGA zumeist schon im technischen Systementwurf festgelegt werden muss.

Die folgenden Teilabschnitte präsentieren Implementierungsvarianten des Datentyps Floating-Point doppelter Genauigkeit. Die Gründe dafür sind einerseits die häufige Verwendung bei Realisierungen von komplexen Algorithmen der Signalverarbeitung in messtechnischen Systemen und anderseits ist die Unterstützung dieses Datentyps immer noch eine Herausforderung auf FPGAs.

Die Version der Entwicklungsumgebung LabVIEW 2013 enthält für den FPGA-Entwurf vorbereitete Funktionsblöcke beziehungsweise mathematische Operationen zur Berechnung mit Fließkomma-Arithmetik einfacher Genauigkeit. Es gibt auch mehrere OpenSource-VHDL-Bibliotheken und IP-Cores, die zur Implementierung von Operationen in diesem Datenformat eingesetzt werden könnten [OC15][FPL13]. Deswegen war eine der Aufgaben im praktischen Teil der vorliegenden Dissertation die Implementierung von Double-Operationen für einen FPGA. Im Folgenden werden die Möglichkeiten auf Basis von speziell entwickelten Bibliotheken (IP-Cores) und weiterhin mittels eines dafür entworfenen Softcore-Prozessors detaillierter dargestellt. Die alternative Verwendung der beiden Realisierungsarten zeigt die in ZEfIRA vorgesehene Möglichkeit, verschiedene Varianten bezüglich Datenformats, Zeitbedingungen und Ressourcenverbrauchs in Sinne des Findens einer optimalen Lösung zu betrachten.

4.4.1.2 Realisierung der wiederverwendbaren Floating-Point-Bibliothek

Die erste Möglichkeit ist die Implementierung von Gleitkomma-Arithmetik doppelter Genauigkeit in Form einer IP-Bibliothek. Diese wird in der Arbeit Floating-Point-Bibliothek genannt und kann für unterschiedliche Anwendungen flexibel und anpassbar wiederverwendet werden. Für die Dissertation wurde diese Variante für FPGAs des Herstellers Xilinx verfolgt und in [GDMF12] mit anderen Realisierungsmöglichkeiten verglichen.

Der FPGA-Hersteller Xilinx stellt einen CoreGenerator zur Verfügung, der das automatische Erstellen von unterschiedlichen IP-Cores für spezielle FPGA-Familien ermöglicht [Xil14]. Es existiert auch die Möglichkeit die Operatoren variabel zu generieren (z.B. mit unterschiedlicher Eingangs- beziehungsweise Ausgangsdatengenauigkeit, Ressourcenverbrauch, Laufzeit der Ausführung). Bei der Entwicklung von für die messtechnischen Aufgaben notwendigen Operationen doppelter Genauigkeit werden die Verhältnisse zwischen den benötigten FPGA-Ressourcen und den Latenzen untersucht und sind einstellbar. Dabei bewirken die kleineren Latenzzeiten den höheren Ressourcenverbrauch. Diese Einstellbarkeit erlaubt die Implementierung von spezialisierten Operationen, die für einen konkreten Algorithmus optimal angepasst werden. Z.B. wählt man für die Realisierung von Regelungsalgorithmen die dabei zu verwendenden Gleitkomma-Operatorvarianten

aus, die ein optimales Verhältnis zwischen Zeit- und Ressourcenverbrauch unter Berücksichtigung der benötigten Tastperiode besitzen. Als Ergebnis der Core-Generierung werden ein HDL-Code und eine Netlist-Datei (*.ngc–File) erzeugt. Diese werden in unterschiedliche Entwicklungsumgebungen integriert [Xil14]. Die entsprechenden Eigenschaften wie Ressourcenverbrauch und Ausführungszeit jeder entworfenen Operation werden bei der Ressourcen- und Zeitanalyse benötigt und müssen dafür entsprechend der Verwendung im Prozess ZEfIRA dokumentiert werden (s. Tab. A.1 im Anhang A.3).

Beispielhaft wird hier die Integration in die Entwicklungsumgebung LabVIEW dargestellt. Zur Einbindung von im Xilinx CoreGenerator generierten Operatoren für Gleitkomma-Arithmetik müssen die Knoten “HDL-Interface-Node” oder “IP-Integration Node” (ab LabVIEW10) verwendet werden [LV15]. Die integrierten Bibliotheken werden in LabVIEW als Unterprogramme abgespeichert und können zu den Nutzer-Funktionen hinzugefügt werden. Die Abbildung 4.17 zeigt die entwickelte Floating-Point-Bibliothek. Diese ist in allen LabVIEW-Versionen für verschiedene Xilinx-FPGAs wiederverwendbar.

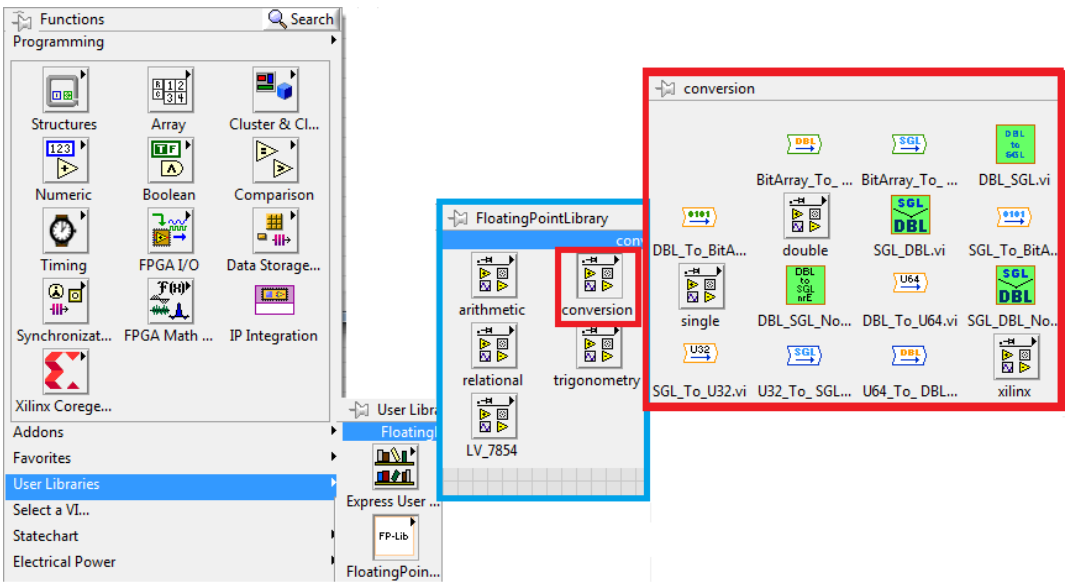


Abbildung 4.17: Überblick zur Floating-Point-Bibliothek in LabVIEW

Es wurde eine Klasse von mathematischen Operationen und Funktionen realisiert, die für die messtechnische Anwendung relevant sind. Dazu gehören mathematische Operationen (Addition, Subtraktion, Multiplikation, Division und Quadratwurzel) und logische Vergleichsfunktionen („gleich“, „nicht gleich“, „größer“, „größer gleich“, „kleiner“, „kleiner gleich“). Außerdem werden auch weitere mathematische Funktionen und Konvertierungsoperationen zwischen verschiedenen Datentypen gebraucht. Zu den mathematischen Funktionen gehören Vergleichsfunktionen und trigonometrische Funktionen (z.B. Sinus, Cosinus), die sowohl bei der Realisierung von

Signalverarbeitungsaufgaben als auch bei der prototypischen Untersuchung der Messkette nötig sind. Außerdem spielen die Konvertierungsfunktionen eine wichtige Rolle (rechte Seite der Abb. 4.17). Diese ermöglichen z.B. die Umwandlung des abgetasteten Signals aus Integer in Floating-Point.

Im Anhang A.3 (Tabelle A.1) sind die entwickelten Floating-Point-Bibliotheken mit deren relativem Ressourcenverbrauch auf dem FPGA Virtex-5-LX110 und den erzielten Ausführungszeiten dargestellt. Die Operationen Division und Quadratwurzel benötigen im Vergleich mit den anderen mehr Ressourcen und bedeutend längere Ausführungszeiten. Es wird deshalb empfohlen bei der Analyse und Realisierung von Signalverarbeitungsalgorithmen, diese Operationen falls möglich durch andere zu ersetzen.

4.4.1.3 Realisierung eines FPGA-basierten Softcore-Prozessors

Die Implementierung von komplexen Algorithmen bei der Verwendung von IP-Cores und VHDL-Bibliotheken mit Fließkomma-Arithmetik ist auf modernen FPGAs sehr ressourcenaufwendig. Dazu kann die direkte Umsetzung von mathematischen Operationen ineffizient sein, das heißt z.B. im Zeitverbrauch nicht besser als eine Software-Lösung. Es ist auch möglich, dass eine auf dem FPGA nicht realisierbare Lösung entsteht. Die nachfolgend beschriebene Implementierungsvariante mit einem speziellen Softcore-Prozessor legt ein steuerflussorientiertes Scheduling zu Grunde. Für die Einschätzung der Effektivität ist die Betrachtung des resultierenden Zeit- und Ressourcenverbrauchs möglich. Es wird ein systematischer und konsequenter Weg zur Verwendung von Steuerfluss-Programmen entwickelt, um eine Ergänzung beziehungsweise Alternative zur Parallelität bei angestrebter besserer Ressourcenauslastung auf dem FPGA durchzuführen. Dieser Prozessor liegt dann als wiederverwendbare Realisierung komplett als Modul für den FPGA vor. Vorteile der Verwendung von softwarebasierten Prozessoren bestehen in der höheren Flexibilität und größeren Funktionalität. Außerdem hat es sich bei den praktischen Realisierungen gezeigt [MKGP13], dass bei geeigneter Softcore-Architektur der Zeitverbrauch der Operationen sogar geringer sein kann, als bei der Verwendung von direkt umgesetzten Funktionsblöcken.

Es existiert eine Vielzahl von OpenSource-Prozessoren [OC15], die sich für Signalverarbeitungsaufgaben und für Berechnungen mit Fließkomma-Arithmetik eignen und die als editier- und veränderbare Vorlage gedacht sind. Sie können im Idealfall auch ohne spezielle Anpassungen direkt verwendet werden. Diese Softcore-Prozessoren sollen für ein großes Spektrum an Aufgaben zur Verfügung stehen und enthalten deshalb eine relativ breite Funktionalität, die für die konkrete Anwendung eventuell unzureichend ist. Zumeist verbrauchen diese OpenSource-Prozessoren dadurch mehr Ressourcen (z.B. Chipfläche, elektrische Leistung etc.), als vergleichbare, für spezielle Anwendungen entwickelte und optimierte Softcore-Prozessoren. In den Masterarbeiten von Rozova [Roz13] und Kirchhoff [Kir14] wurde dieses u.a. anhand unterschiedlicher bekannter Lösungen

aus der Literatur genauer untersucht und beschrieben. Weitere dort ermittelte Aussagen sind, dass die Änderbarkeit und die Unterstützung von Fließkomma-Arithmetik doppelter Genauigkeit bisher nur eingeschränkt verfügbar sind.

Aus diesen Gründen wurde im Fachgebiet Rechnerarchitektur und Eingebettete Systeme ein eigenes Konzept einer konfigurierbaren Prozessorarchitektur für messtechnische Signalverarbeitungsaufgaben [PKMG11][DPF12] entworfen. Dabei wird eine effiziente, für diese Aufgabenklasse anpassbare Realisierung erreicht. Der Grundsatz der Entwicklung dieser angepassten Prozessoren im vorgesehenen Applikationsfeld besteht darin, die Anforderungen, welche die Leistungsfähigkeit von DSPs und die Flexibilität von Mikrocontrollern kombinieren, zu erfüllen. Dabei mussten die in den Signalverarbeitungsaufgaben benötigten Funktionen mit Fließkomma-Arithmetik doppelter Genauigkeit realisiert werden. Als wichtigste Leistungseigenschaft wurde eine möglichst geringe und vor allem planbare Latenz angesehen [DPZF13].

Die Abbildung 4.18 stellt die allgemeine konzeptionelle Struktur des entwickelten Softcore-Prozessors dar.

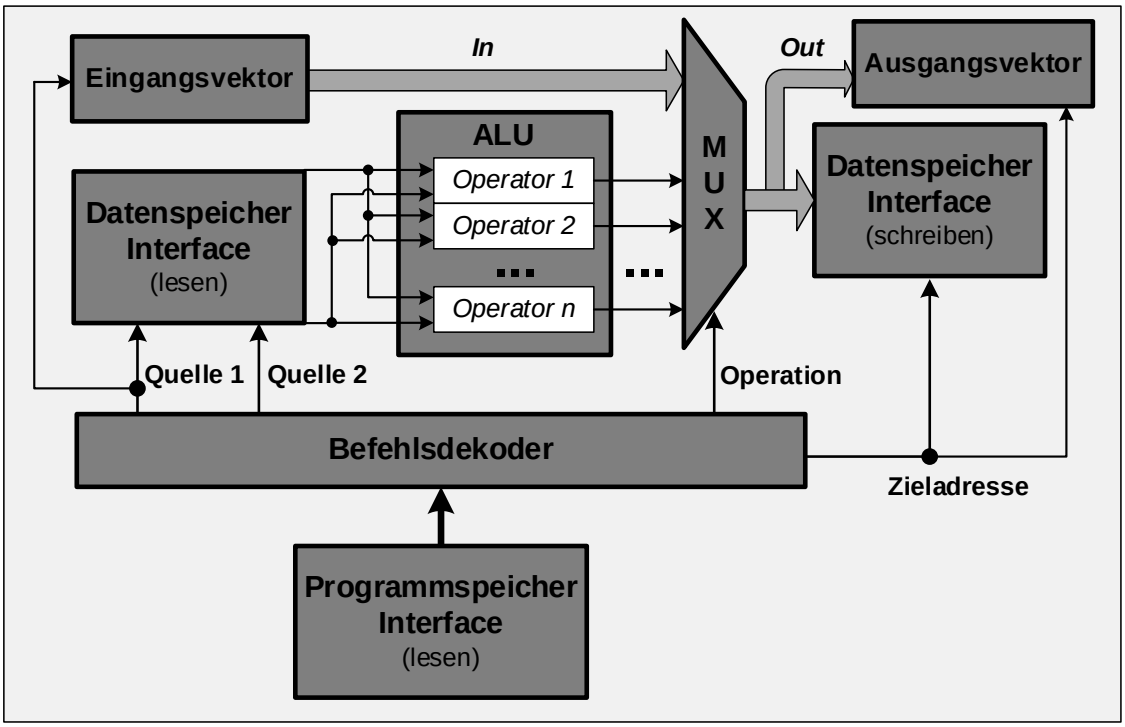


Abbildung 4.18: Struktur des Softcore-Prozessors

Der Eingangsvektor dient dem Einlesen von Sensorwerten oder anderen Eingangsdaten. Mit jedem Takt wird die aktuelle Zeile des Programms, welches sich im Programmspeicher befindet, als Befehlscode an den Befehlsdekoder gegeben. Abhängig von der Art des Befehls werden beide Operanden gleichzeitig aus dem Datenspeicher oder dem Ein-

gangsvektor gelesen. Im Datenspeicher können bei jedem Befehlsschritt zwei Quelloperanden adressiert werden. Diese Operanden werden an das Rechenwerk (*Arithmetic Logic Unit* - ALU) übergeben, das alle benötigten Fließkomma-Operatoren in paralleler Anordnung enthält. Weiterhin existieren zwei Lese- und ein Schreiboperator, die zwei Eingangsoperanden bereitstellen (Datenspeicher Interface bzw. Eingangsvektor lesen) und ein Ergebnis schreiben (Datenspeicher Interface schreiben beziehungsweise Ausgangsvektor). In der ALU arbeiten alle vorhandenen Operationen gleichzeitig mit denselben Eingangsoperanden, unabhängig vom Befehl. Durch den Block MUX (Multiplexer) wird entsprechend dem dekodierten Befehl das Ergebnis der benötigten Operation ausgewählt und zum Schreiben beziehungsweise Ausgeben weitergeleitet.

Es wurde eine Harvard-RISC-Architektur mit fünfstufiger Pipeline umgesetzt. Die Stufen der Pipeline sind: *Befehl lesen* (Ausführung 1 Takt), *Befehl dekodieren* (Ausführung 1 Takt), *Operanden lesen* (Ausführung 1 Takt), *Operation ausführen* (Ausführung je nach Operation 1 bis n Takte) und *Ergebnis speichern* (Ausführung 1 Takt). Durch die Pipeline im Zusammenwirken mit einem Codeconverter, der unter anderem eine Befehlsreihenfolgeänderung nach dem Out-of-Order-Prinzip durchführt, wird sichergestellt, dass in jedem Prozessortakt eine mathematische Operation gestartet und gleichermaßen eine beendet werden kann. Das gilt auch, wenn Operationen mehrere Takte zur Ausführung benötigen. Solche Operatoren wurden voll pipelinisiert ausgeführt. [DPF12][Sa13]

Die in der ALU realisierten Fließkomma-Operatoren wurden mit Hilfe einer verfügbaren IP-Bibliothek entworfen, die sowohl einfache, als auch zusammengesetzte Operatoren bietet. Diese sind in VHDL beschrieben und können immer wieder geändert beziehungsweise an die konkrete Aufgabestellung angepasst werden. Details dazu wurden im Abschnitt 4.4.1.2 beschrieben.

In den Arbeiten [Roz13] und [Kir14] ist der Befehlssatz des entwickelten Softcore-Prozessors dargestellt. Dabei wurden in Anpassung an die vorgesehenen Informationsverarbeitungsaufgaben nicht nur mathematische Grundoperationen, wie z.B. Addition und Multiplikation realisiert, sondern auch spezielle, häufig in der Signalverarbeitung verwendete Funktionen, wie Konvertierungen von Datentypen, transzendente Funktionen (Sinus, Cosinus, Exponentiell), und Datentransfers, auch zwischen Softcore- und FPGA-Datenspeicher, umgesetzt (s. Anhang A.3: Tab. A.2 und A.3). Allgemein ist es wichtig zu bemerken, dass in dem Konzept des Softcore-Prozessors keine Sprungbefehle vorgesehen wurden, um eine Vereinfachung der Ablaufsteuerung zu bewirken und bestimmte Verzögerungsquellen zu beseitigen. Das leitet sich aus den Aufgaben der Signalverarbeitung in der Messtechnik ab, die einen festgelegten einmaligen Berechnungsablauf pro Abtasttakt benötigen. Bei jedem Abtasttakt wird das ganze Programm neu gestartet und ausgeführt. Bei Notwendigkeit von kurzen bedingten Programmteilen werden die im Befehlssatz realisierten bedingten MOV-Befehle verwendet.

Ein wichtiger Schwerpunkt der Entwicklung von Softcore-Prozessoren liegt in der begleitenden Verifikation [Sch13]. Die Verwendung der Simulation als Verifikationsmethode ist aufgrund der Komplexität der Systeme nicht vollständig. Dadurch können zwar Fehler entdeckt werden, das ermöglicht aber nicht den Nachweis der Korrektheit des kompletten Systemverhaltens. Die Methoden der formalen Verifikation ermöglichen letzteres. Durch ein mathematisches Modell lässt sich die Existenz interessierender Eigenschaften beweisen.

In der Arbeit von Schulze [Sch12] wurde eine Methodik zur formalen Verifikation von Softcore-Prozessoren entwickelt. Am Beispiel des Oregano 8051 Mikrocontrollers gliederte sich die Verifikation in angepasste Gruppen: Befehlsabarbeitung (inklusive Interrupts), Timer und serielle Schnittstelle. Für jede Gruppe wurde eine separate Datei erstellt, die den PSL (Property Specification Language)-Verifikationscode beinhaltet. Bei der Aufteilung in Verifikationsgruppen wurde das System speziell strukturiert, um Überlappungen und Lücken zwischen den einzelnen Gruppen zu vermeiden. Als Ergebnis wurden für dieses Beispiel Bugs bezüglich der Befehlsausführung (CJNE-Carry-Flag Bug), des Interruptverhaltens (INT-RETI Bug & Interruptschutzbug) und des Timers (Timer Mode 0 Bug & Timer Pipeline Bug) entdeckt und konnten dadurch behoben werden.

Es ist weiterhin wichtig für den entwickelten und hier dargestellten Softcore-Prozessor, den speziell dafür realisierten Codekonverter nutzen zu können. Dieser wird gebraucht, um die Umsetzung von programmtechnischen Realisierungen mit dem Softcore-Prozessor in dessen Maschinencode bei Anwendung des oben genannten Out-of-Order-Prinzips durchzuführen. Für den Umsetzungsvorgang des Codekonverters werden die verfügbaren Assemblerbefehle mit ihren Eigenschaften und Parametern in einer Liste nach vorgegebenen Regeln (Abb. 4.19: *Konfiguration*) abgelegt. Der Codekonverter sucht bei der Übersetzung für jeden Befehl im Programm entsprechend der Datenabhängigkeiten eine geeignete Position im Maschinenprogramm. Dabei werden folgende Schritte für jeden Befehl ausgeführt (s. Abb. 4.19):

1. *Prüfen der Datenabhängigkeiten:* Abhängigkeiten zwischen den Befehlen werden bestimmt (Lesen nach Schreiben, Schreiben nach Lesen, Schreiben nach Schreiben)
2. *Suchen nach freien Codezeilen im Zielprogramm:* Unter Berücksichtigung der Ausführungszeit jedes Befehls und entsprechend der bestimmten Datenabhängigkeiten werden die Warteplätze automatisch aufgefüllt
3. *Einfügen des betreffenden Befehls:* Durch die Umrechnung der Operandenadressen werden die Codezeilen modifiziert

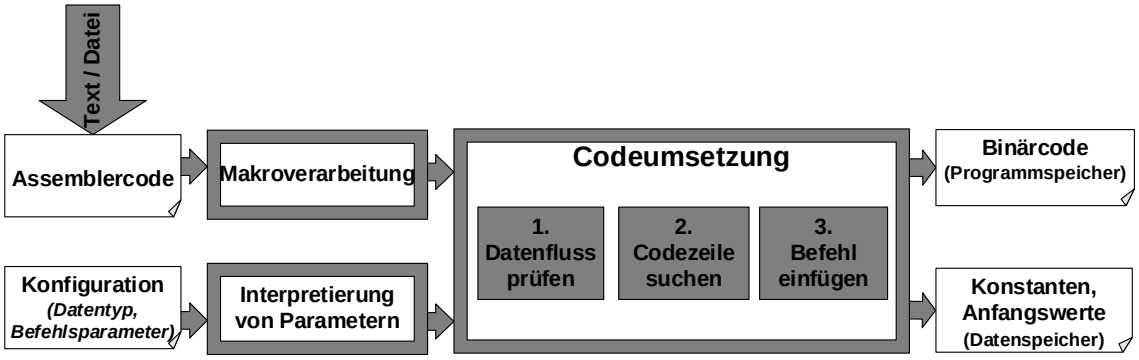


Abbildung 4.19: Ablauf der Codekonvertierung

Während der Programmübersetzung werden Zeit- und Speicherbedarf ermittelt und eventuelle Syntaxfehler detektiert. Als Ergebnis der Konvertierung werden Inhalte des Programm- und Datenspeichers (letzteres für Anfangswerte und Parameter) generiert. Diese werden in die Softcore-bezogenen Speicherbereiche auf dem FPGA eingespielt.

Das beschriebene Konzept des Softcore-Prozessors wurde in zwei Varianten entwickelt. Die erste Realisierung, der Softcore LiSARD (LabVIEW-integrated Softcore Architecture for Reconfigurable Devices), wurde vorwiegend für die prototypische Entwicklung entworfen [PKMG11]. Die Implementierung erfolgte vollständig in der Entwicklungsumgebung LabVIEW. Hier wurden die Architektur des Prozessors und dessen Umgebung, sowie die Realisierung des Codekonverters mit LabVIEW-Komponenten umgesetzt [DPF12][Roz13]. Eine weitere Realisierung, der Softcore ViSARD (VHDL-integrated Softcore Architecture for Reconfigurable Devices), wurde im Kontext dieser Dissertation unter dem Gesichtspunkt einer produzierbaren und kostengünstigeren Realisierung entwickelt (ZEfIRA: dritter W-Prozess). Hierzu enthält dieser Prozessor Funktionen zur Abschaltung von nichtbenutzten Operatoren, hauptsächlich zur Reduzierung des Stromverbrauchs. Die Umsetzung des Prozessors ViSARD und dessen Testumgebung erfolgte in der ISE Design Suite [Xil14] der Firma Xilinx mittels VHDL. Dadurch besteht die Möglichkeit zur Validierung mittels des integrierten Simulator ISim. In der Arbeit von Sauer [Sa13] wurde ein Hardware-Testkonzept für diesen Softcore-Prozessor untersucht und realisiert. Die Masterarbeit von Kirchhoff [Kir14] stellt die vervollständigte Testumgebung und Erweiterungen zum Softcore dar. Eine zusätzliche Arbeit wurde für die Implementierung des für den ViSARD-Core speziell angepassten Codekonverters durchgeführt. Der ViSARD Core Codegenerator wurde mittels Python realisiert [Kra14].

Im Kapitel 5 wird die Verwendung des Softcore-Prozessors LiSARD bei der prototypischen Entwicklung für die konkrete Applikation in der Wägetechnik beschrieben. Dabei werden gezielte Einschränkungen und Vereinfachungen im Sinne nicht benötigter Funktionen umgesetzt. Dazu werden auch spezielle, für die prototypische Entwicklung wichtige Eigenschaften und Funktionen berücksichtigt.

4.4.2 Plattformspezifische Ressourcenanalyse

Bei der plattformspezifischen Modellierung werden gewisse Charakteristika der Ziel-Hardware-Plattform (hier FPGA) bezüglich der Verfügbarkeit von Berechnungselementen, Ein-/Ausgangs- und Speicherschnittstellen, Interprozessor- oder Intermodulkommunikation in Betracht gezogen. Im idealen Fall wird die Spezifikation direkt in ein plattformspezifisches Komponentenmodell verfeinert und jede Funktion nach einer exakten Relation auf eine beziehungsweise mehrere Logikkomponenten abgebildet. Dieses ermöglicht eine maximal ausgenutzte vollständig parallele Verarbeitung auf dem FPGA. In der Regel ist aber die direkte Abbildung eines komplexen Datenflussmodells auf die FPGA-Struktur ohne Optimierungen nicht möglich. Es müssen mehrere Optimierungsschritte und -iterationen durchgeführt werden, die die Realisierbarkeit bezüglich der Ressourcen und den gestellten Anforderungen als Ziel haben. Dabei wird zumeist nicht formal optimiert, sondern auf Grund von Erfahrungen der Entwickler, a-priori-Wissen und durch die Bewertung zur Verbesserung des Ressourcenverbrauchs getroffener Entscheidungen.

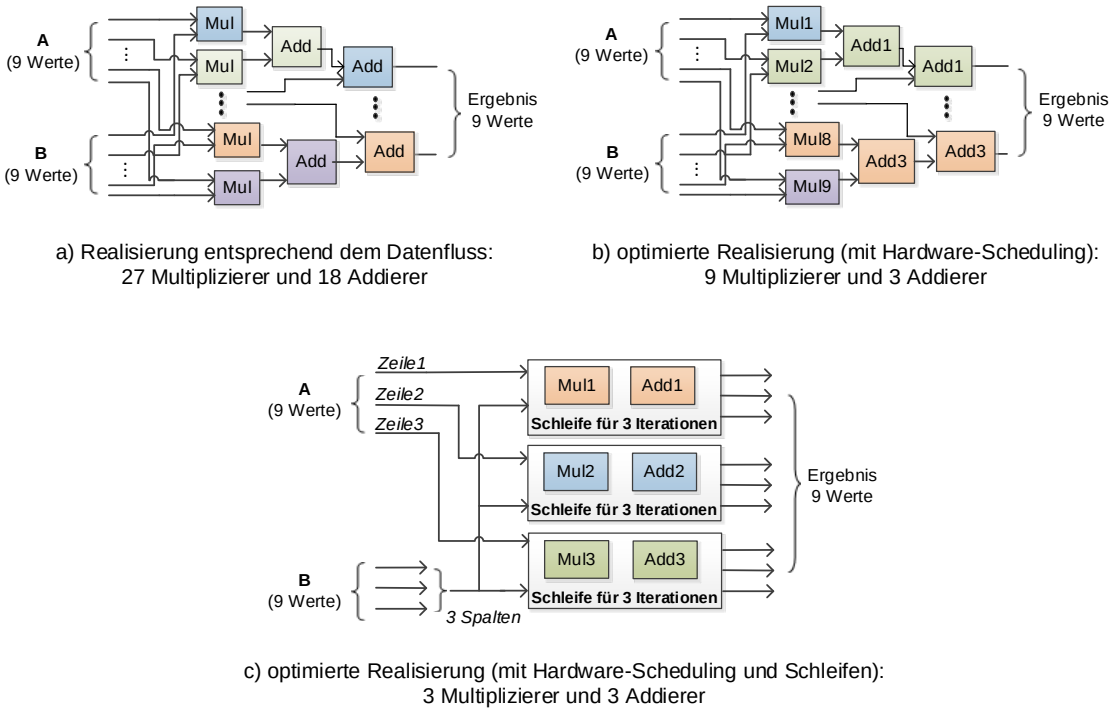
Der modellbasierte verfeinerte technische Entwurf soll die Aktivitäten zur Strukturoptimierung unterstützen. In diesem Fall werden die benötigten Algorithmen in Bezug auf verwendete mathematische Operationen untersucht. Es können Zustände oder Parameter ermittelt werden, die sich nicht ändern. Dabei ist z.B. das Eliminieren von Multiplikationen mit Null oder Eins und von Additionen mit Null sinnvoll. Als Beispiel dieser Optimierung ist die Realisierung eines Kalman-Algorithmus [MKGP13] zu nennen, der zahlreiche Matrixmultiplikationen und einige Vektoroperationen in Fließkomma-Arithmetik enthält. Die Modellierung des Datenflusses ergab die Verwendung von 208 Additionen, 280 Multiplikationen, 17 Subtraktionen und einer Division. Die Untersuchungen in dem genannten Applikationsbeispiel (Kalman-Filter als Beobachter im Regelungssystem der NPMM) zeigten, dass die komplexe Struktur des Filters auf 90 Multiplikationen, 85 Additionen, eine Subtraktion und eine Division reduziert werden konnte.

Ein weiterer Punkt des iterativen Optimierungsprozesses betrifft den effizienten Ressourcenverbrauch der einzelnen Elementarfunktionen in der Struktur.

Eine Möglichkeit besteht in der mehrfachen Verwendung von denselben Ressourcen, indem die Abbildung mehrerer gleicher Funktionsblöcke auf einen Hardware-Operator stattfindet. In diesem Fall werden lokale Steuerflüsse erzeugt, die jeweils Sequenzen für die Verwendung desselben Hardware-Elements an unterschiedlichen Stellen des Berechnungsweges realisieren, ohne dass die Parallelisierung von Teilen des Algorithmus dabei ausgeschlossen wird. Zusätzlich können auch mehrere gleichartige Blöcke eine Sequenz von Operationen auf derselben Struktur definieren, was weitere Ressourceneinsparungen auf Kosten von Parallelität ermöglicht. Insgesamt ist dabei zu beachten, dass bei mehrfacher Nutzung derartiger Hardware-Elemente notwendigerweise zusätzliche Logik für das

Umschalten der Datenflüsse beziehungsweise Parameter notwendig wird. Das führt zu einem durch die zusätzliche Logik verursachten Ressourcenverbrauch. Bei der sequenziellen Verwendung von Hardware-Strukturen entstehen andere Zeitverhältnisse gegenüber der parallelen, die bei der noch folgenden Zeitanalyse berücksichtigt werden. Die genannte Technik ist als *Resource-Sharing* bekannt und wurde in der Dissertation von Müller in MATLAB/Simulink umgesetzt [Mü12]. In anderen Entwicklungsumgebungen, z.B. LabVIEW, wird diese Technik weitgehend automatisch auf Elementaroperatorebene unterstützt (unter der Einstellung „reentrant“ lassen sich VIs mehrfach platzieren und mit „non-reentrant“ - einmal platzieren).

Als Beispiel für die verschiedenen Varianten werden Realisierungen einer Matrizenmultiplikation AxB (3x3) ohne (*a*) und mit Hardware-Scheduling (*b*, *c*) in der Abbildung 4.20 dargestellt.



a) Realisierung entsprechend dem Datenfluss:
27 Multiplizierer und 18 Addierer

b) optimierte Realisierung (mit Hardware-Scheduling):
9 Multiplizierer und 3 Addierer

c) optimierte Realisierung (mit Hardware-Scheduling und Schleifen):
3 Multiplizierer und 3 Addierer

Abbildung 4.20: Realisierungsbeispiel zur Multiplikation zweier 3x3-Matrizen

Die Variante (*a)* zeigt die direkte Umsetzung der Matrizenmultiplikation auf dem FPGA entsprechend dem Datenflussmodell. In diesem Fall werden alle benötigten Operatoren (Multiplizierer und Addierer) nach einer (1:1)-Relation auf die Logikkomponenten abgebildet. Dabei wird die vollparallele Verarbeitung maximal genutzt. Das führt unter anderem zum günstigsten Zeitverhalten. Die bezüglich des Ressourcenverbrauchs optimierten Realisierungen (*b, c*) unterscheiden sich in der Berechnungsstruktur.

Die Variante (*b*) stellt ebenfalls die elementweise Realisierung der Matrizenmultiplikation aber mit Hardware-Scheduling (mehrfacher Verwendung von denselben Ressourcen)

dar. Für die Operatoren (hier Multiplizierer), die in der Variante (*a*) parallel auf dem Chip rechnen können, ist diese Variante auf Kosten der Parallelität ressourcen-sparsamer. Dadurch wird aber die Ausführungszeit verlängert (in diesem Beispiel verdreifacht). Die weitere Optimierungsmöglichkeit (*c*) bezieht sich auf eine veränderte Struktur. Die Operationen werden nicht mehr elementweise berechnet, sondern in Vektoren (für die Matrizen A und B: Spalten bzw. Zeilen) geteilt und mit Schleifen realisiert. Dadurch entsteht eine günstigere Gesamtstruktur und es werden Ressourcen gespart. Die Fähigkeit des FPGA zur parallelen Verarbeitung wird dabei eingeschränkt.

Eine durchgeführte Realisierung eines Kalman-Filters [GDMF12] ergab Ergebnisse für unterschiedliche Varianten auf einem FPGA. Auf Grund der komplexen Struktur des Filters wurden bei der Umsetzung Optimierungen mit Hardware-Scheduling und Schleifen benutzt. Es wurden die zwei besten von vier Realisierungen ermittelt: platzoptimierte Umsetzung (verbraucht 70% der Ressourcen des benutzten FPGA) und zeitoptimierte Umsetzung (verbraucht 90% der Ressourcen des benutzten FPGA). Beide Realisierungen entsprechen der gestellten Anforderung: Laufzeit des Algorithmus unter 10 µs.

Weiteres Potenzial zur Ressourcen-Optimierung bietet die Verwendung eines steuerflussorientierten Rechenkerns, der in Form eines Softcore-Prozessors für FPGA entwickelt wurde. Diese Variante ist im Abschnitt 4.4.1.3 dargestellt und stellt letztendlich auch eine spezielle Variante des Resource-Sharing dar.

Eine formale Methode der Ressourcenanalyse kann die Bewertung der verschiedenen Varianten unterstützen. Für die vorliegende Dissertation wurde in der Masterarbeit von Kozlachkova [Ko13] untersucht, wie Petri-Netze dafür verwendet werden können. Der zu realisierende Algorithmus wird dabei mit einem Petri-Netz modelliert und in einem geeigneten Tool simuliert (z.B. s. Abb. 4.21).

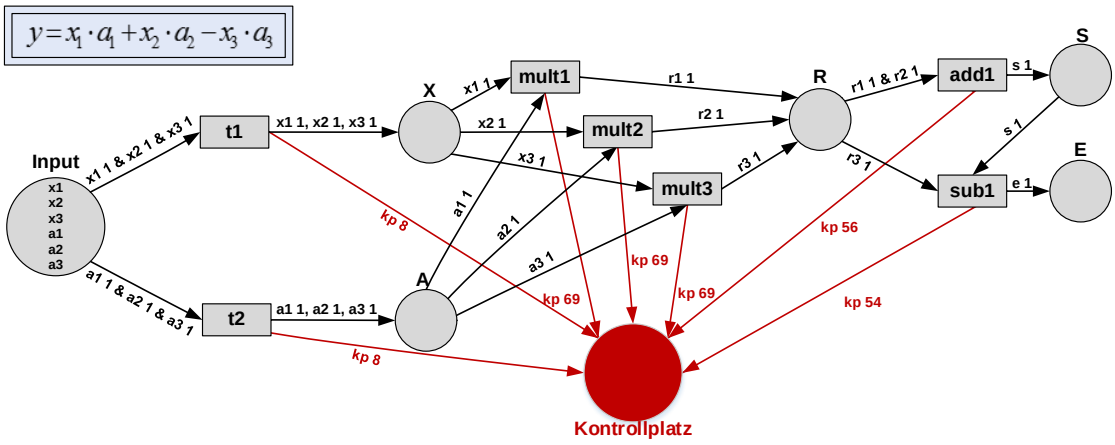


Abbildung 4.21: Beispiel eines Petri-Netzes zur Ressourcenanalyse

Das abgebildete Beispiel demonstriert die Berechnung einer Gleichung, in der drei parallel ausgeführte Multiplikationen, eine Addition und eine Subtraktion benötigt werden. Das Petri-Netz für den Datenfluss und den darin implizit enthaltenen Steuerfluss des Algorithmus wird durch Elemente ergänzt, mit deren Hilfe die Analyseergebnisse dargestellt werden können. In diesem Zusammenhang können die benötigten Ressourcen abhängig von den verwendeten Datentypen und Ablaufreihenfolgen ermittelt werden. Dabei wird der Ressourcenverbrauch der Einzeloperationen durch spezielle Kantenbewertungen beschrieben. Z.B. benötigt der Multiplikator 6,9% (s. Anhang A.3: Tab. A.1), deswegen liefert die entsprechende Kante 69 Marken zum Kontrollplatz. Indem dieser spezielle Kontrollplatz mit einer Kapazität entsprechend der Anzahl der verfügbaren Ressourcen (hier 1000 bei der Normierung 100%) versehen wird, lässt sich Prüfung auf Konfliktsituationen im Netz auf Ressourcenkonflikte schließen. Wenn ein Nachkonflikt in Bezug auf den Kontrollplatz entsteht, werden im modellierten System mehr Ressourcen gebraucht, als zur Verfügung stehen. Außerdem hat der Kontrollplatz die Rolle eines Indikators, welche Anzahl an Ressourcen in jeder Etappe der Ausführung verwendet wird. Zur Erweiterung kann die Kapazität des Kontrollplatzes auf unendlich gesetzt werden, um das Netz auf Beschränktheit zu untersuchen. In diesem Fall kann man aus der Schranke der Markenanzahl im Kontrollplatz auf das Maximum der notwendigen Ressourcen für den modellierten Algorithmus und die gewählte Realisierung schließen.

4.4.3 Plattformspezifische Zeitanalyse

Die im technischen Systementwurf durchgeführte Zeitanalyse (Abschnitt 4.3.4) wird als Ausgangspunkt beim plattformspezifischen Entwurf benutzt. Auf Grund von Verfeinerungen und der Festlegung der verwendeten Plattform kann eine genauere Abschätzung des zeitlichen Verhaltens im Eingebetteten System durchgeführt werden. In diesem Fall sollen die Ergebnisse der Datentyp- und Ressourcenanalyse herangezogen werden, da diese Änderungen im Zeitverhalten bewirken. Die Zeitanforderungen können im Gegenzug auch auf die verwendbaren Datentypen und benötigten Ressourcen Einfluss haben.

Die Spezifika von FPGA-basierten Eingebetteten Systemen bestehen darin, dass eine exakte Zeitvalidierung erst nach Synthese der Funktionen und Routing auf der Chipfläche möglich ist. Bei der modellbasierten Untersuchung kann man aber eine Abschätzung des Zeitverhaltens der FPGA-Realisierungen durchführen. Dafür müssen bestimmte Kenntnisse über die Ausführungszeit von zu verwendenden Komponenten beziehungsweise Teilsystemen zur Verfügung stehen. Hierzu wird empfohlen, beim verfeinerten Entwurf ergänzend formale Verifikationsmethoden zur Abschätzung des Zeitverhaltens zu benutzen und die vollständige Zeitanalyse der gesamten Realisierung nach der Ausführung auf dem FPGA-Chip durchzuführen.

Zur Ermittlung von Informationen über Ausführungszeiten von Komponenten der Realisierung können werkzeugunterstützte Mechanismen der modellbasierten Entwicklung

verwendet werden. Z.B. steht in der Entwicklungsumgebung LabVIEW die Funktion „Tick Count“ zur Verfügung, die eine taktbezogene Zeitmessung von verwendeten Operationen beziehungsweise Teilsystemen ermöglicht (s. Beispiel Abb. 4.22).

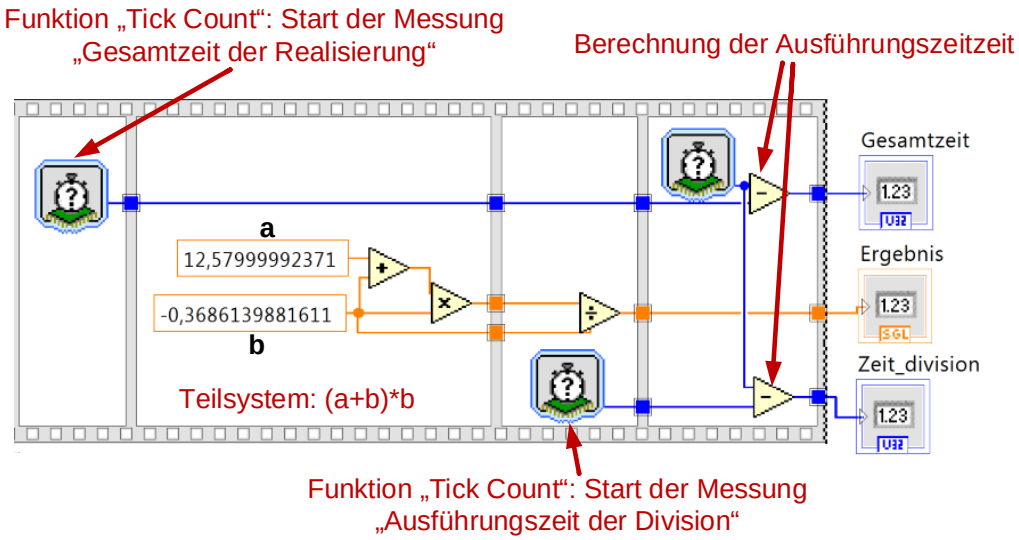


Abbildung 4.22: Beispiel der Zeitmessung in LabVIEW

Anders ist es bei der Generierung von neuen Bibliotheken (z.B. Floating-Point-Bibliotheken: Abschnitt 4.4.1.2). In diesem Fall werden die Eigenschaften (z.B. Datentyp, Ressourcenverbrauch, Laufzeit der Ausführung) bei der Erstellung der Operation festgelegt und können bei weiteren Untersuchungen benutzt werden.

Für den plattformspezifischen Entwurf wurde eine formale Methode für die Zeitanalyse unter Verwendung von Zeitintervall-Petri-Netzen entwickelt. Diese wird in Weiteren beschrieben. Dadurch lassen sich mehrere Zeitbedingungen überprüfen [DH01]:

- ✓ Ermittlung von kürzester und längster Zeitdauer für zusammengesetzte Funktionen;
- ✓ Einhaltung spezifizierter Zeitschranken (deadlines);
- ✓ benötigte Minimal- und Maximalzeiten für Intervalle einer beziehungsweise mehrerer Transitionen, bei denen die Schaltsequenz eine geforderte Dauer nicht überschreitet;
- ✓ Minimal- und Maximalzeiten für Intervalle einer beziehungsweise mehrerer Transitionen, in denen ein bestimmter Zustand (erreichbare Markierung) im Netz erreicht werden kann (zeitabhängige Lebendigkeit).

Auf Basis der modellierten Funktion wird deren Steuerfluss mit Zeitbedingungen durch Elemente eines zeitbewerteten Petri-Netzes dargestellt, und jeder Transition, die einer bestimmten Operation entspricht, wird ein Zeitintervall als Verzögerung zugeordnet.

Dadurch wird das Schalten der Transition von der Ausführungszeit der Operation abhängig. Die Verwendung eines Intervalls ermöglicht die Modellierung für den Fall, dass die Ausführungszeit in Grenzen variabel beziehungsweise nicht exakt bekannt ist.

Die Abbildung 4.23 stellt ein Beispiel der Modellierung einer Gleichung für zwei Ausführungsvarianten dar. Die Zeitintervalle der mathematischen Operation sind in Takten entsprechend der Tabelle A.1 (Anhang A.3) modelliert. In diesem Fall sind Minimum und Maximum gleich, das heißt, es existiert genau eine bekannte Zeit. Das ist in der Eigenschaft dieser FPGA-Operationen begründet, mit einer festen Taktanzahl zu arbeiten.

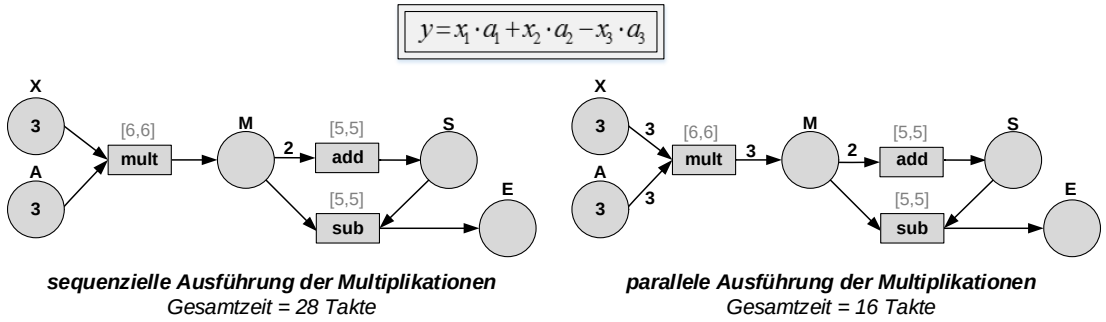


Abbildung 4.23: Beispiel des modellierten Petri-Netzes zur Zeitanalyse

Die Masterarbeit von Kozlachkova [Ko13] untersuchte für die vorliegende Arbeit die unterschiedlichen Möglichkeiten zur Zeitanalyse mit Petri-Netzen. Hierzu wurden die Tools TINA [T15] und PENECA Chromos [PC15] für die Modellierung benutzt. Ein Mangel dieser Werkzeuge war bei der Analyse von komplexen Algorithmen festzustellen. Da in diesem Fall Untersuchungen mit gefärbten zeitbehafteten Petri-Netzen notwendig waren, konnten die obengenannten Tools nicht eingesetzt werden. TINA unterstützt zwar Zeitintervalle, aber keine gefärbten Petri-Netze. Letzteres ist bei PENECA Chromos gegeben, wobei hier nur feste Zeitwerte möglich sind. In diesem Zusammenhang wurde deshalb nur das prinzipielle Vorgehen der formalen Zeitanalyse beschrieben.

4.5 Modulentwurf und FPGA-Implementierung

In diesem Abschnitt wird der Implementierungsprozess auf FPGAs, angepasst und eingeordnet in ZEfIRA, erläutert. Er behandelt zwei näher betrachtete Modulgruppen, die in den untersten Schritten des W-Prozesses umgesetzt werden: FPGA-Komponenten und Softcore-Programme. Im W-Modell der Prototypenentwicklung entsprechend der Abbildung 3.2 werden diese gemeinsam als FPGA-Module bezeichnet. Diese Partitionen des Eingebetteten Systems werden, wie in der Abbildung 4.24 gezeigt, weitgehend getrennt behandelt. Der FPGA-Modul-Test erfolgt dann wieder weitgehend gemeinsam.

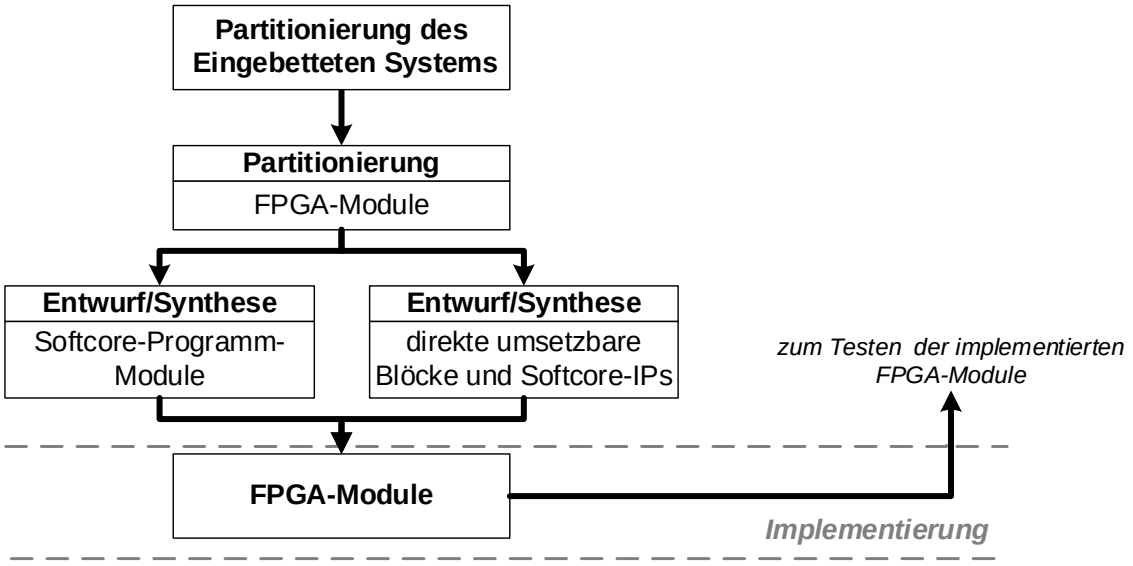


Abbildung 4.24: Implementierung der FPGA-Module

Nach dem verfeinerten technischen Entwurf wird das plattformspezifische Modell in einen implementierbaren Code umgewandelt (s. Abschnitt 2.4.3). Die folgende Implementierung für die Zielplattform übernehmen danach herstellerspezifische Werkzeuge (z.B. Compiler, Synthese-Tools). Entwicklungsumgebungen, wie MATLAB/Simulink und LabVIEW, enthalten spezielle Mechanismen beziehungsweise Ergänzungen zur automatischen Codegenerierung. Diese sind bei der prototypischen Entwicklung zu empfehlen, da dadurch Entwicklungsaufwand und –zeit erheblich verkürzt werden.

Im Sinne einer zertifizierbaren Entwicklung (hier vor allem bei der produzierbaren Lösung: dritter W-Prozess) ist die Verwendung von speziellen zertifizierten Codekonvertern und Compilern notwendig. Sinnvollerweise kann schon bei der Prototypenentwicklung darauf geachtet werden, dass derartige Werkzeuge verfügbar sind und eventuell schon im zweiten W-Prozess eingesetzt werden. Eigenentwickelte Codekonverter können so aufgefasst werden, dass sie äquivalent zur Software im Messgerät zu behandeln sind. Damit gelten für sie die Empfehlungen der PTB zu dieser (s. Abschnitt 2.3.2).

Die Abbildung 4.25 stellt den Ablaufprozess einer FPGA-Implementierung dar (Erweiterung der Abb. 2.9, s. Abschnitt 2.4.3.2). Die tatsächliche Implementierung auf dem FPGA beginnt mit der Abbildung der Logik in Form von HDL-Beschreibungen auf spezifische Hardware-Elemente und deren Verbindungen (engl. *Mapping*). Während der Place-Route-Etappe geschieht die nachfolgende Platzierung und Verdrahtung der Logik auf der Chipfläche. Außerdem werden bei der Implementierung neben den Hardware-Beschränkungen die vordefinierten Nutzerbeschränkungen (z.B. bestimmte Latenzanforderungen, Taktfrequenzen, Platzierung von Funktionen auf der Chipfläche) berücksichtigt. Entwurfsbegleitend werden Validierungsschritte im Ablaufprozess durchgeführt. Nach der Verhaltenssimulation mit HDL-Simulatoren können basierend

auf den Netzlisten funktionale Simulationen auf Gatterebene durchgeführt werden. Die eigentliche Simulation des Zeitverhaltens auf dem Chip, die Timing Simulation, wird erst mit den gerouteten Netzlisten mit Timing möglich. Als Ergebnis der Implementierung wird ein Bitfile des implementierten Entwurfs generiert, das die Konfigurierung des FPGAs ermöglicht.

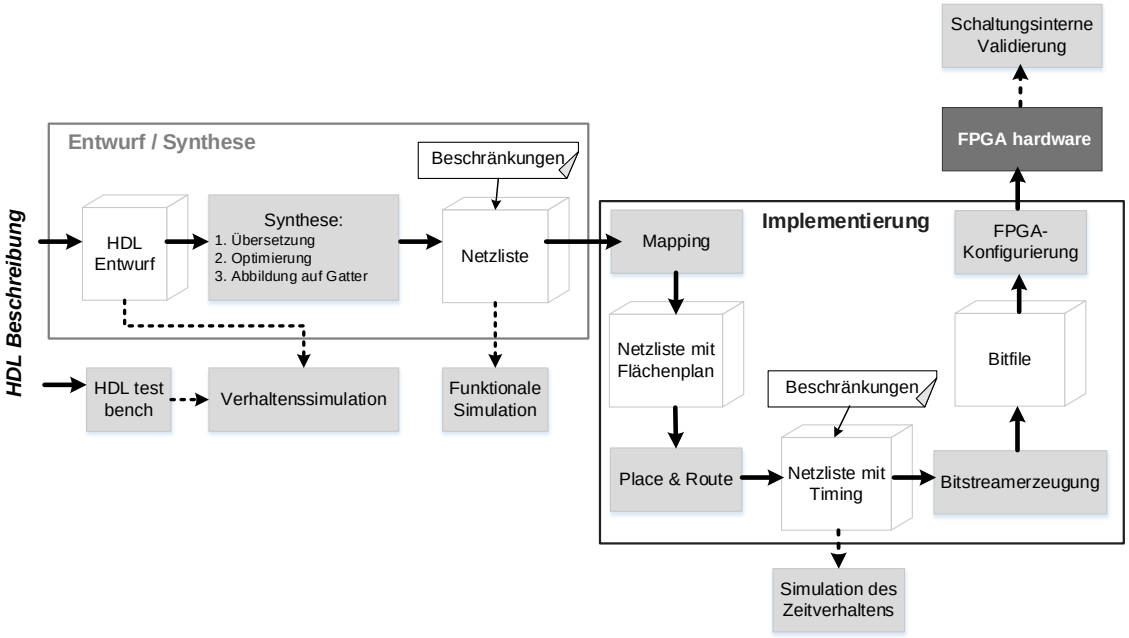


Abbildung 4.25: Implementierungsprozess für FPGAs

Bei der Realisierung von Algorithmen in Form von Programmen für Softcore-Prozessoren werden eine Reihe von zusätzlichen Schritten und Werkzeugen notwendig. Dieses war einer der näher zu untersuchenden Aspekte der vorliegenden Dissertation und wird im Weiteren genauer betrachtet. Die Verwendung eines speziellen Softcore-Prozessors bei der Umsetzung von komplexen Algorithmen der Signalverarbeitung bedingt, wie im Abschnitt 4.4.1.3 schon erwähnt, die Realisierung von geeigneten Codeumsetzern. Die ermöglichen eine automatische Generierung aus dem Modell in den Softcore-Maschinencode. Die Abbildung 4.26 zeigt den für die vorliegende Arbeit entwickelten Ablauf und die darauf aufbauend implementierten Teilkomponenten.

Der Hauptschritt der Umsetzung des spezifizierten Algorithmus zur Ausführung auf dem Softcore-Prozessor ist die Realisierung eines entsprechenden Assembler-Programms. Hierzu werden zwei grundsätzliche Wege betrachtet: manuelle und automatische Erstellung des Codes. Bei der manuellen Erstellung werden die modellierten Algorithmen in Form eines speziellen Assembler-Programms durch den Entwickler realisiert. In diesem Fall müssen Entwicklungszeit und -aufwand im Entwurfsprozess zusätzlich berücksichtigt werden. Dagegen hat die automatische Codeerstellung mehrere Vorteile. Neben der

schnellen Entwicklung ist die Reduzierung von Fehlern zu nennen, die durch den Entwickler verursacht werden können. In der Arbeit von Ölschlegel wurde ein automatischer Codegenerator auf Basis einer Bibliothek zur Erzeugung eines Assemblerprogramms aus einem MATLAB/Simulink-Modell entwickelt [Öl15]. Eine andere Variante bezieht sich auf die Verwendung der Programmiersprache C/C++, die eine erweiterte Schnittstelle zwischen dem modellierten Algorithmus und dem Assemblercode darstellt. In diesem Fall wird der modellierte Algorithmus in C-Code konvertiert (eine Funktion zur C-Code-Generierung ist in Entwicklungswerkzeugen wie MATLAB/Simulink oder LabVIEW verfügbar). Danach übernimmt ein entwickeltes Programm die Übersetzung in den speziellen Assemblercode des Softcores. Dieses wurde in C++ als Standalone-Programm entworfen und implementiert.

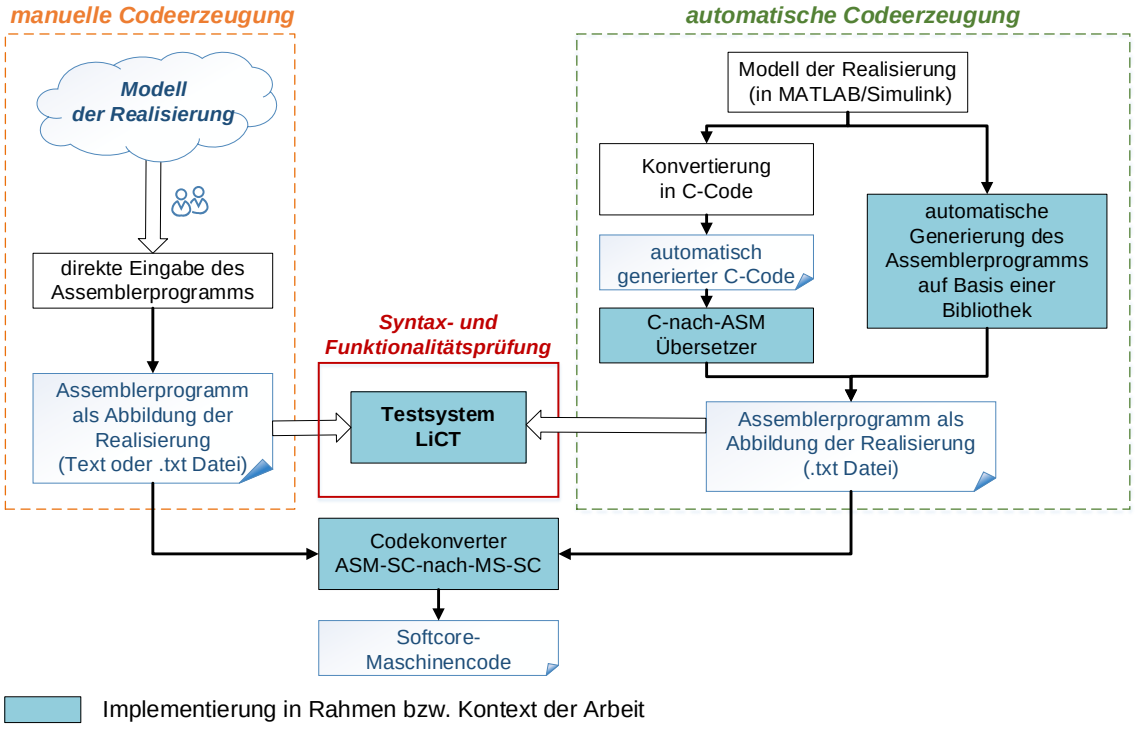


Abbildung 4.26: Werkzeugkette zur Erzeugung von Softcore-Code

Die weitere Umsetzung des Assemblerprogramms ist die Aufgabe des Codekonverters ASM-SC-nach-MS-SC (Assembler-Softcore nach Maschinensprache-Softcore, s. Abschnitt 4.4.1.3). Für die zukünftigen Arbeiten zu einer praktisch anwendbaren Werkzeugkette wäre es sinnvoller, die obengenannten Schritte in einer Entwicklungsumgebung zu realisieren und zu nutzen.

Eine wichtige Rolle bei der Realisierung von Algorithmen für den Softcore-Prozessor spielen die durchführbaren Testaktivitäten. Besonders bei der manuellen Erstellung von Softcore-Programmen können verschiedene Entwicklungs- und Syntaxfehler auftreten.

Für diese Dissertation wurde in der Forschungsarbeit von Nagaraj [Nag14] ein Testsystem (LiCT: LiSARD Code Tester) mit integriertem Codesimulator entwickelt und vollständig implementiert, das zur Syntax- und Funktionalitätsprüfung benutzt werden kann. Es enthält die Möglichkeit zur Eingabe des Assemblerprogramms und der Testdaten aus einer Datei oder über die Benutzeroberfläche. Gleiches gilt in umgekehrter Richtung für die Ausgabe von Testergebnissen. Zusätzlich bietet das Testsystem Möglichkeiten zur Prüfung des Assemblerprogramms einer Funktion mittels des Vergleichs mit deren mathematischer Darstellung. Die Gleichungen werden bei der Eingabe auf syntaktische Zulässigkeit überprüft und über einen Interpreter abgearbeitet. Dafür kann man die Gleichung einer Funktion und das entwickelte bzw. erzeugte Assemblerprogramm mit entsprechenden Testdaten untersuchen, um die Berechnungsergebnisse zu vergleichen. Dabei können prinzipiell auch Testdatensätze verwendet werden, die durch den Testdatengenerator erzeugt werden.

4.6 Testen des realisierten und implementierten Systems auf den behandelten Abstraktionsebenen

Im Prozess ZEfIRA spielen die entwurfsbegleitenden Testaktivitäten eine besondere Rolle.

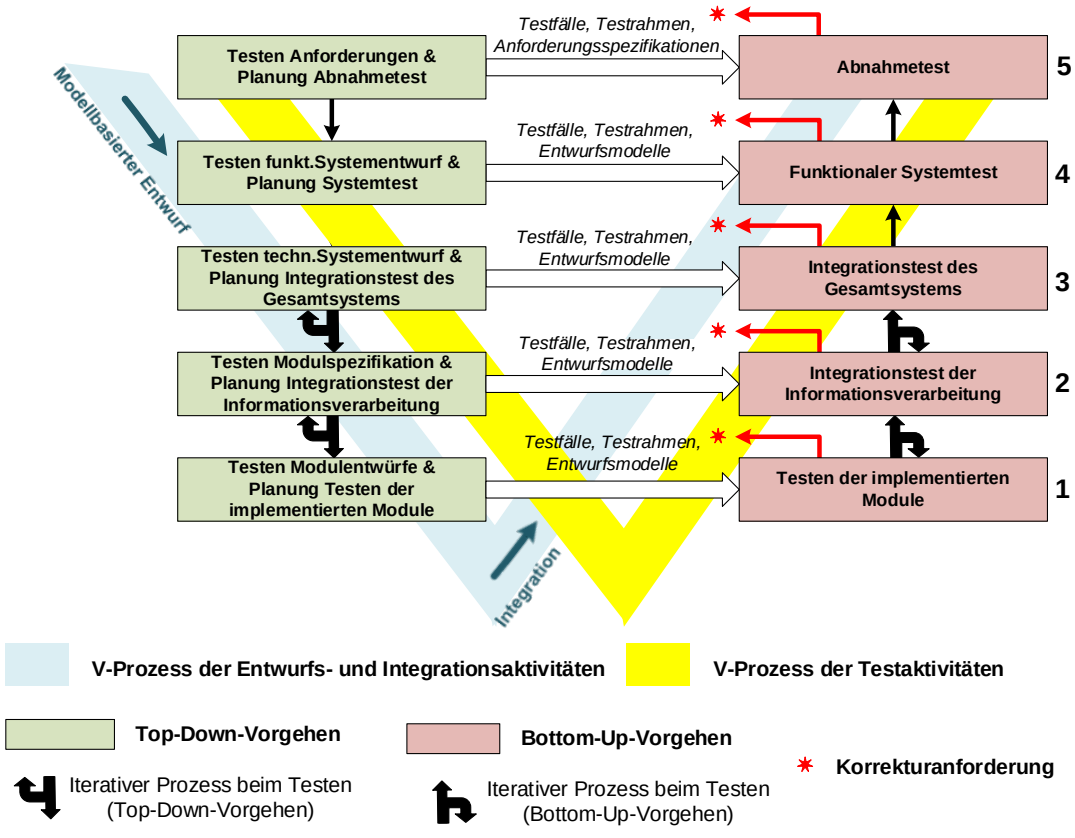


Abbildung 4.27: Testaktivitäten im W-Modell nach ZEfIRA

Das Testen und die Planung der Prüfung des entworfenen Systems für den Bottom-Up-Verlauf auf den unterschiedlichen Abstraktionsebenen des Top-Down-Entwurfes dienen zur späteren effektiven Testdurchführung (s. Abb. 4.27). Die bei dem modellbasierten Entwurf durchgeführten Tests mit festgelegten Zielen (Etappen des Top-Down-Vorgehens) werden nach der Realisierung beziehungsweise Implementierung des Systems in der Umgebung mit realen Bedingungen validiert (Etappen des Bottom-Up-Vorgehens). In diesem Fall werden die unterschiedlichen Aktivitäten vorbereitet und in Form von Testfällen und Testrahmen übergeben. Zum einen legen diese die Teststrategie fest, also welche Methoden beim Testen für welche implementierten Funktionen und auf welchem Abstraktionsniveau verwendet werden. Zum anderen werden die Testaktivitäten bezüglich des zeitlichen und organisatorischen Ablaufs geplant. Hierbei wird in den Top-Down-Etappen des Entwicklungsprozesses bereits geplant, in welcher strukturellen Zuordnung welche Tests stehen und in welchem logischen und zeitlichen Umfang sie durchgeführt werden. Eine weitere wichtige Aktivität ist die frühzeitige Vorbereitung der Testumgebung, um die diesbezüglichen Erkenntnisse aus den entwurfsbegleitenden Tests des Top-Down-Ablaufs zu übernehmen. Die spezifizierten Testfälle, -rahmen und -ideen müssen entsprechend ZEfIRA dokumentiert werden.

Die Besonderheiten des im Kapitel 3 beschriebenen Konzeptes bestehen darin, dass bei der Prototypenentwicklung (W-Prozess: “Prototyp“) die Testdurchführung in dem realisierten und implementierten System auf das zukünftige Testen der produzierbaren Lösung orientiert wird. Bei Prototypen wird das Testen meistens durch die Entwickler selbst durchgeführt, wobei die späteren Tester beteiligt werden können. Die Testetappen beziehen sich auf die Testumsetzung, die Protokollierung und die nachfolgende Auswertung. Die Nachweise unter Berücksichtigung der Eichfähigkeit und Metrologischen Sicherheit sind in diesem Fall wichtige und spezifische Bestandteile der Überprüfung, die eine besondere Betrachtung erfordern. Beispielsweise gilt das für einen geeigneten Test aller Aspekte der Manipulationssicherheit. Anderseits unterstützt der 3W-Prozess die Übernahme von Testverfahren und Testergebnissen aus dem Vorläufersystem. Das bedeutet, dass im Bottom-Up-Vorgehen die Kenntnisse über Testumgebung und –durchführung genutzt werden können, aber auch die Nutzung der zur Verfügung gestellten Einheiten zum Testen (z.B. experimentelle FPGA-Boards, analoge Geräte) vorgesehen ist. Auch für Testgeräte und –umgebungen gelten die Forderungen und Richtlinien der Eichbehörden und anderer Organisationen sinngemäß.

In der Abbildung 4.27 ist der Fall der Korrekturanforderung in jeder Abstraktionsetappe zur besseren Übersicht nicht vollständig dargestellt. Dieser entspricht dem Sprung aus dem rechten Ast des W-Modells (Testen) in den linken (Entwurf, Korrektur). Wenn bei der Testdurchführung festgestellt wurde, dass das realisierte beziehungsweise implementierte System nicht fehlerfrei ist, sollen die Fehler auf der Hierarchieebene, in der sie entstanden sind, in den Entwurfsmodellen korrigiert werden. Ab dieser Stelle ist ein erneutes

Durchlaufen des W-Prozesses notwendig. Das ist insbesondere auf Grund der Tatsache, dass ein zertifizierbarer Entwicklungsprozess (Anforderung an ZEfIRA) entstehen soll, gegeben. Im Weiteren wird der Verlauf beim Testen im Bottom-Up-Vorgehen detaillierter beschrieben.

In der ersten Teststufe (**1**) werden die nach der Realisierung bzw. Implementierung oder Programmierung erstellten Module einem systematischen Test unterzogen. Kennzeichnend für diese Etappe ist, dass jeweils ein einzelnes Modul isoliert überprüft wird. In diesem Fall werden die internen Aspekte validiert. Als Testbasis werden die den Modulen zugrundeliegenden spezifisch entworfenen Modelle herangezogen. Die wichtigste Aufgabe dieser Teststufe ist die Prüfung der Korrektheit und Funktionalität der implementierten Module mittels einer Reihe von Testfällen, wobei jeder Testfall eine bestimmte Teilfunktionalität abdecken soll. Neben diesem sollen Robustheit und nicht funktionale Eigenschaften überprüft werden, die die Qualität der Module maßgeblich beeinflussen (z.B. Effizienz und Wartbarkeit [SL12]).

Der Nachweis der Robustheit verwendet als Ausgangspunkt Testmethodenaufrufe, Daten und Sonderfälle, die entsprechend der Spezifikation unzulässig sind. In diesem Fall wird die Reaktion der Module nach definierten Ausnahmekriterien untersucht. Außerdem werden zur Überprüfung der Manipulationssicherheit spezielle Angriffe gegen die Systemteile durchgeführt. Das gilt auch allgemeiner für alle Varianten unzulässiger Möglichkeiten des Einflusses auf die Messwertermittlung. Im Weiteren wird dieser Aspekt auch in den höheren Hierarchieebenen betrachtet werden müssen. Ein gutes Beispiel auf dem untersten Niveau ist die Untersuchung, inwieweit die Komponente der Verschlüsselung korrekt und wirksam ist.

Bei der Prüfung der Effizienz der Module werden solche Teilaspekte wie z.B. Ausführungszeit, Latenzen, Ressourcen- und Speicherverbrauch entsprechend der beim Modulentwurf spezifizierten Forderungen betrachtet.

Die Prüfung der Wartbarkeit stellt Aspekte wie Modularität, Strukturierung und Kommentierung des Software-Codes sowie Verständlichkeit und Aktualität der erstellten und übergebenen Dokumentation in den Vordergrund. In diesem Fall wird für den hier als Schwerpunkt betrachteten Fall auch überprüft, ob die Dokumentation entsprechend der Forderungen von Eichbehörden und anderen Organisationen erstellt wurde.

Es ist weiterhin zu beachten, dass eine nachträgliche Korrektur von später beim Einsatz auftretenden Fehlern durch einen Update-Mechanismus auf Grund der Eichanforderungen nicht möglich ist. Im Folgenden wird dieser Aspekt auch in den höheren Hierarchieebenen betrachtet werden müssen. Er führt insbesondere dazu, dass umfangreichere und möglichst die meisten Fälle abdeckende Tests notwendig werden.

Die nächste Teststufe (**2**) bezieht sich auf die Integration der entwickelten und überprüften Module für das Informationsverarbeitungssystem. In diesem Zusammenhang muss

getestet werden, ob die Teilkomponenten richtig miteinander funktionieren. Dabei wird vorausgesetzt, dass die Module auf Grund der durchlaufenen Teststufe (**1**) und eventuellen Korrekturzyklen weitgehen korrekt sind. Das Hauptziel ist die Validierung des Zusammenwirkens der zu integrierenden Module und der Schnittstellen zwischen ihnen. In dieser Etappe können vor allem Fehler entdeckt werden, die auf Grund des fehlerhaften Zusammenwirkens der bereits getesteten Module entstehen beziehungsweise erst sichtbar werden:

- fehlerhafte Struktur unter Benutzung fehlerfreier Module;
- fehlerbehaftete Module, wobei die Fehler beim Modultest in der darunterliegenden Ebene nicht gefunden wurden und erst durch das Zusammenwirken in Erscheinung getreten sind;
- fehlerhafte Datenübermittlung zwischen Komponenten (z.B. falsche Datenformate, fehlende Daten und Signale). Sie weist auf funktionale Fehler der Module, Protokollfehler oder inkompatible Schnittstellenformate hin;
- Zeitprobleme, die vor allem durch zu geringen Datendurchsatz, unzureichende Kapazität bei zu hoher Last und beim Datentransfer (falscher oder verspäteter Zeitpunkt) auftreten;
- widersprüchliche oder fehlinterpretierte Spezifikation von Modulen und Struktur.

Die Ursachen für die auftretenden Fehler sind auf dieser Abstraktionsebene zumeist komplex und oft nicht trivial aus den entstandenen Fehlerbildern interpretierbar. Insbesondere können auch Kombinationen von Fehlern auftreten. Dabei müssen die Schnittstellen von Modulen, die zur Berechnung der Messwerte beitragen, wiederum auf Grund der Forderung der Eichfähigkeit und Metrologischen Sicherheit besonders untersucht werden.

Da beim Zusammenwirken der Module ein enger Zusammenhang zu den komplexeren Algorithmen besteht, sollten hier die Testergebnisse für die Eichbehörde in der dieser Behörde zu übergebenden Dokumentation der Algorithmen geeignet dargestellt werden.

Zur Fehlersuche ist es zumeist sinnvoll geeignete Testsysteme zu benutzen. Das sind oft Fremdsysteme oder zugekaufte Komponenten. Analog zu den Möglichkeiten der Simulation bei der modellbasierten Entwicklung können z.B. zeitliche Parameter gemessen und angezeigt werden. Ein geeignetes Messgerät dafür wäre ein Logikanalysator mit Mixed-Signal-Funktionalität. An dieser Stelle ist die Verwendung von Testsystemen und –komponenten des Vorläufersystems sinnvoll. Wie bereits oben genannt gelten für die Testsysteme die Forderungen der Eichbehörden und anderen Organisationen sinngemäß.

Der Integrationsprozess von Modulen aus unterschiedlichen Partitionen, z.B. FPGA-Modul und Softcore-Modul, kann iterativ durchgeführt werden (s. Abb. 4.27). Das ist typi-

scherweise dann der Fall, wenn Teilmodule in beiden Partitionen enthalten sind. Sie können zumeist nur im Zusammenwirken sinnvoll getestet werden. Dabei erfolgt ein Vorgehen wie in der Testetappe **1** teilweise auch in der Etappe **2**. Gleiches tritt im Weiteren auch in der Testetappe **3** auf und wird dort weiter ausgeführt.

Die dritte Testetappe (**3**) bezieht sich auf die Integration des Eingebetteten Systems (Informationsverarbeitung) in das Einbettende System. Im Zuge der Integration werden die Einzelkomponenten schrittweise zu größeren Einheiten des Gesamtsystems zusammengesetzt. Es können dabei mehrere Integrationsstufen auftreten, die Testobjekte unterschiedlicher Größe betreffen. Hier, wie auch in der zweiten Etappe, werden Schnittstellenfehler und das Zusammenwirken von Komponenten im Gesamtsystem überprüft. Als Testbasis werden z.B. die Architektur des Gesamtsystems, schnittstellübergreifender Daten- und Steuerfluss, oder Anwendungsfälle, oder auch elektronische Verbindungen zwischen dem Einbettenden und dem Eingebetteten System herangezogen. Dabei wird davon ausgegangen, dass auf Grund der davor liegenden Testetappe für die beiden Teilsysteme die Aspekte der Eichfähigkeit und Metrologischen Sicherheit einzeln, so weit möglich, untersucht wurden.

Entsprechend der definierten und durchgeführten Tests bei der modellbasierten Entwicklung werden vorwiegend die gleichen Testdatengeneratoren und Testsignale auch in der realen Umgebung verwendet.

Integriert in das Testen des Gesamtsystems sollte hier die Wirksamkeit aller Maßnahmen in früheren Entwurfs- und Testetappen zur Gewährleistung der geforderten Metrologischen Sicherheit untersucht werden. Beispielweise kann erst auf der aktuellen Testetappe realistisch getestet werden, ob die in der Informationsverarbeitung implementierten Algorithmen und das Einbettende System im Zusammenhang dem Verhalten von dessen Modell und der Modelle der Informationsverarbeitung entsprechen. Das ist wiederum ein Schritt zur Erhöhung der Metrologischen Sicherheit.

Außerdem können zusätzliche Geräte herangezogen werden, die unter anderem auch Abläufe beziehungsweise Funktionen der realen Umgebung des Gesamtsystems nachbilden und dabei die Durchführung von definierten Testfällen ermöglichen. Z.B. wurden in der Dissertation von Baumgartl [Ba15] zwei Ansätze zur Validierung einer EMK-Waage erläutert: mittels Einsatz eines elektromagnetischen Lastwechslers, beziehungsweise durch pneumatisches Abheben und Aufsetzen von Gewichtsstücken. In beiden Fällen ist es möglich, eine vordefinierte und reproduzierbare Kraft mit entsprechendem Zeitverhalten auf der Waageschale zu erzeugen. Diese Ansätze wurden für die vorliegende Arbeit auf ihre Eignung untersucht und verwendet (s. Abschnitt 5.2.6.2). Das entspricht der Übernahme eines aus dem Vorläufersystem zur Verfügung gestellten Testsystems.

Der funktionale Systemtest (**4**) findet nach dem abgeschlossenen Integrationstest des Gesamtsystems statt. Das Gesamtsystem wird bei der Überprüfung aus der durch den Auftragnehmer interpretierten Perspektive des Auftraggebers, eventuell des späteren Anwenders und der Eichbehörden und anderen Organisationen betrachtet. In dieser Etappe wird validiert, ob und wie gut das entworfene System den im Pflichtenheft gestellten funktionalen und nicht funktionalen Anforderungen entspricht. Außerdem können Fehler und Mängel in den Anforderungen noch nachträglich identifiziert werden. Eine weitere wichtige Aufgabe der zertifizierbaren Entwicklung ist die Überprüfung der Dokumentation auf Vollständigkeit und Klarheit entsprechend den Forderungen der Behörden und Organisationen (z.B. PTB). Dazu gehört auch die Anwenderdokumentation (z.B. Systemhandbücher, Bedienungsanleitungen).

Die letzte Teststufe (**5**) ist der Abnahmetest, den der Auftraggeber durchführt und für den er verantwortlich ist [SL12]. Normalerweise ist diese Etappe erst für das produzierte Erzeugnis vorhanden. In ZEfIRA wird aber gefordert, diese Teststufe auch im W-Prozess „Prototyp“ durchlaufen zu lassen. Dabei spielen die späteren Erzeugnisentwickler bzw. die für die Anforderungsdefinition zuständige Struktureinheit die Rolle des Auftraggebers. In diesem Zusammenhang können die Testaktivitäten des dritten W-Prozesses „Produktentwicklung“ vorbereitet und untersucht werden. Als Testkriterien gelten die in der Anforderungsspezifikation festgeschriebenen Abnahmekriterien. Auch Erfüllung von relevanten gesetzlichen Vorschriften, Normen oder Sicherheitsrichtlinien wird dabei in der betrachteten Domäne überprüft. Als Testbasis dienen alle Dokumente oder Informationen, die das Testsystem aus Anwendersicht und aus der Sicht der Eichbehörden beschreiben (Anwender- und Systemanforderungen, Formulare, Berichte, Gesetze, Vorschriften, Richtlinien). Im Gegensatz zu den anderen Testetappen wird diese in der Abnahmeumgebung des Auftraggebers durchgeführt. Wegen der unterschiedlichen Testumgebungen entstehen eventuell Probleme, die vorher nicht beobachtet wurden beziehungsweise nicht beobachtet werden konnten. Außerdem sind die Prozeduren zur Integration in die Anwenderumgebung und die dafür notwendige Konfiguration des Systems als Teil der Abnahme zu überprüfen. Ein Beispiel kann sein: Bei der Entwicklung einer Wägezelle wurde eine spezielle Belastungsvorrichtung benutzt, und der Anwender untersucht darüber hinaus deren Einsatz in einer dafür vorgesehenen Verpackungsmaschine.

Nach den Tests des Auftraggebers muss in der Domäne der eichfähigen Messtechnik eine weitere Testetappe durchlaufen werden. Diese wird vom Auftraggeber initiiert und von den Eichbehörden und eventuell anderen betroffenen Organisationen (z.B. FDA) durchgeführt. In diesem Fall werden nicht mehr alle spezifizierten Eigenschaften, sondern nur eine gezielte Auswahl aus diesen, die in Gesetzen, Richtlinien und Vorschriften festgelegt sind, untersucht. Als Beispiel aus der Wägetechnik kann für das Eichen gelten, dass für die Einhaltung der Messunsicherheit nur exemplarisch mit einigen Eichgewichten getestet wird.

5 Nachweis der Anwendbarkeit von ZEfIRA

Der in der Arbeit entwickelte und in den vorherigen Abschnitten beschriebene Prozess ZEfIRA wurde durch das im Kapitel 3 dargestellte 3W-Modell systematisiert und formalisiert. Zum Nachweis seiner Anwendbarkeit für Entwicklungen in der Anwendungsdomäne der Messtechnik werden jetzt ein Projekt aus der dynamischen eichfähigen Wägetechnik und ausgewählte Projektteile von der Entwicklung einer Nanopositionier- und Messmaschine (NPMM) genutzt. Diese Messsysteme benötigen komplexe und hochleistungsfähige Informationsverarbeitungsfunktionen, wie z.B. Filter-, Regelungs-, Steuerungs- und Bahnplanungsalgorithmen (letztere bei NPMM). Wesentliche Einsatzziele der eingebetteten Echtzeitverarbeitungssysteme sind beispielsweise ein oder mehrere geschlossene Regelkreise mit komplexen Algorithmen der Zustands- beziehungsweise Störbeobachtung und eine Reihe von weiteren Aufgaben der Informationsverarbeitung. Dabei beeinflusst die Ausführungszeit von Algorithmen direkt die Abtastrate, die Latenzen, die Regelabweichung und die resultierende Prozessqualität der Signalverarbeitung. [Mü12][Am10]

Die eigenen Untersuchungen im Gebiet der NPMM haben gezeigt, dass sowohl Prozess-Controller mit leistungsfähigen Prozessoren als auch Signalprozessoren nicht ausreichen, um die extrem hohen Anforderungen an die Signal- und Datenverarbeitung zu erfüllen. Weiterhin wurden die Arbeiten dadurch erschwert, dass bei den verschiedenen Arbeitsgruppen (Messtechnik, Regelungstechnik, Positioniertechnik, Informationstechnik, Bildverarbeitung) kein gemeinsames systematisches Vorgehen im Rahmen eines definierten Entwicklungsprozesses und eines einheitlichen Modellierungs- und Testablaufes existierte. Dennoch wurde im Transferprojekt des DFG-Sonderforschungsbereiches 622 an der TU Ilmenau ein industrietauglicher Prototyp (die NPMM-200) entwickelt. In diesem Prototyp werden modulare PXI-Hardware-Plattformen mit je mehreren FPGA-Modulen und ihren analogen und digitalen Schnittstellen für Signal- und Datenverarbeitungsaufgaben verwendet. Der Prototyp erfüllt die hohen Anforderungen an die Signalerfassung und die Informationsverarbeitung, die bei einer Messunsicherheit im Nanometerbereich realisiert werden müssen. [Mü12][ZKGA13]

Im vorliegenden Kapitel werden ausgewählte Entwicklungsetappen eines Informationsverarbeitungssystems im Rahmen des ZEfIRA-Prozesses dargestellt. Es wird dabei vor allem ein systematischer Weg der Prototypenentwicklung von Informationsverarbeitungssystemen gezeigt. Als Beispiel eines messtechnischen Systems wird eine elektromagnetische Kraftkompensations-Wägezelle (im Weiteren EMK-Wägezelle genannt) betrachtet, wobei die Anwendbarkeit der meisten Prinzipien und der dargestellten Methoden über diese hinaus geht (z.B. für die NPMM-200). In einer Reihe von Fällen werden auch die Querverbindungen zu den W-Prozessen für das Vorläufersystem sowie die produzierbare Lösung, einschließlich einiger Prozessschritte, betrachtet.

5.1 Beschreibung des Vorläufersystems

Dieser Abschnitt zeigt die Darstellung eines Vorläufersystems (s. Abb. 5.1), das bei der Prototypenentwicklung des Informationsverarbeitungssystems einer EMK-Waage betrachtet wurde. Es geht um eine Wägezelle, die typischerweise in der Lebensmittelindustrie zur Dosierung und Abfüllung sowie zur Kontrollwägung in laufenden Produktionsprozessen (dynamische Wägetechnik) eingesetzt wird [Kr04].

Um Kenntnisse aus dem Vorläufersystem nach dem dargestellten Konzept zu übernehmen, werden Entwicklungsetappen des W-Modells für Prototyp und Produkt dieses Systems teilweise nachgestaltet. Vor allem basierend auf den vorhandenen Dokumentationen, Modellen und Messergebnissen, sowie analysiert durch Interviews, wurde dieser W-Prozess in einigen Schritten durchlaufen. Dessen Ergebnisse werden im W-Prozess bei der Prototypenentwicklung etappenweise berücksichtigt. Hierzu spielen die Anforderungen, die Spezifikationen und die Designentscheidungen des Vorläufersystems eine wichtige Rolle für die Erstellung der neuen Spezifikationslösung. Außerdem kann eine Reihe von Teillösungen wiederverwendet werden.

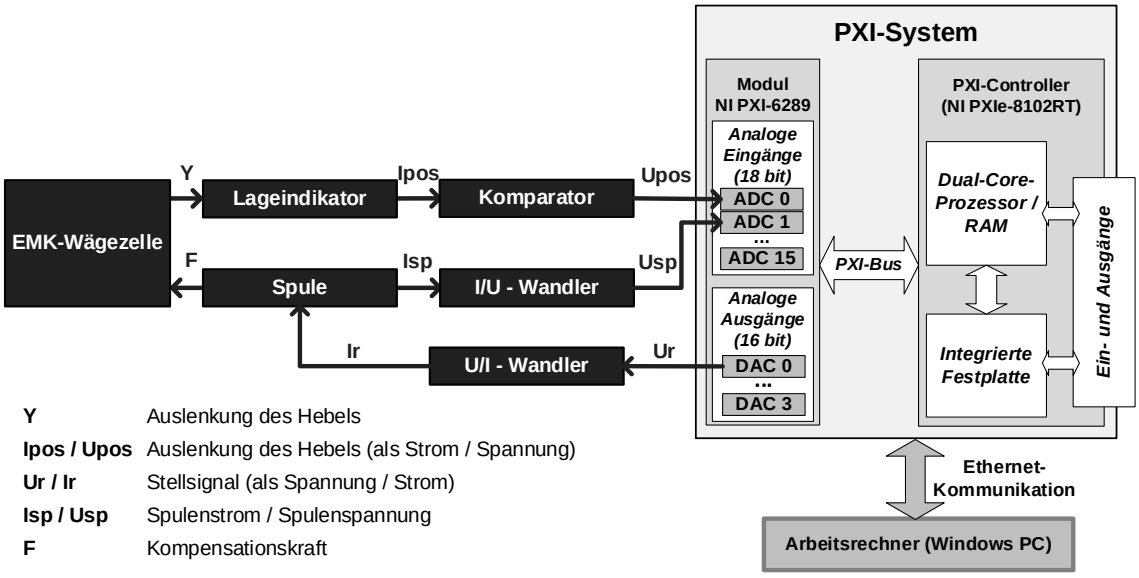


Abbildung 5.1: Schematische Darstellung des Vorläufersystems

Als Implementierungsplattform der digitalen Signalverarbeitung für den Prototyp wurde ein industrielles PXI-System von National Instruments verwendet. Dieses wurde für experimentelle Aufbauten und Untersuchungen eingesetzt und liegt im Kostenrahmen deshalb weit über einer produzierbaren Lösung. Es enthält einen eingebetteten Echtzeitcontroller (NI PXIe-8102 Real Time), der für die Informationsverarbeitungsaufgaben eingesetzt wurde. Dieser Controller umfasst einen Intel Celeron T3100 Dual-Core-Prozessor (1,9 GHz), eine Festplatte, einen RAM (Maximum 2 GB), eine Ethernet-Schnittstelle und

zwei USB-Anschlüsse. Analogschnittstellen zum Messsystem bietet das zusätzliche Modul NI PXI-6289 mit integrierten AD- und DA-Umsetzern. Dieses Modul wird in einen Peripheriesteckplatz im PXI-Chassis eingesetzt. Die erfassten bzw. ausgegebenen Messwerte werden über eine PXI-Backplane transportiert. Zur Messdatenauswertung, Abspeicherung und Darstellung wird zusätzlich ein Arbeitsrechner (PC) verwendet.

Die Aufgabe der Signalverarbeitung besteht aus zwei Teilen: der Regelung und der Bestimmung der Masse. Über den analogen Eingang ADC0 wird die Positionsspannung *Upos* abgetastet und die Messwerte werden über den PXI-Bus zum PXI-Controller transportiert. Die Positionsspannung entspricht der Auslenkung des Hebels, sie wird vom Lageindikator über einen nachfolgenden Komparator (einschließlich analoge Tiefpassfilterung) geliefert. Auf dem Controller wurde ein digitaler Regelalgorithmus realisiert, um den Hebel der Waage in die Nulllage zu steuern. Im Vorläufersystem kommt ein klassischer Regler mit Proportional-, Integral- und Differenzialanteil sowie einem Zeitglied (PID(T1)–Regler) zum Einsatz. Dieser ändert den Strom durch die Spule so lange, bis der Hebel die Nulllage erreicht hat [Ba15]. Zum geschlossenen Regelkreis gehört ein analoger Ausgang DAC0, der das Stellsignal in Form einer Spannung *Ur* ausgibt. Zur Bestimmung der Masse wird ein weiterer analoger Eingang ADC1 benutzt. Es wird der tatsächlich fließende Spulenstrom über einen präzisen Referenzwiderstand gemessen und digitalisiert. Durch die digitale Filterung in Form eines kaskadierten Mittelwertbildners wird die Masse mit der geforderten Auflösung ermittelt und als Ergebnis zur Anzeige ausgegeben [Do08].

Die Algorithmen der Informationsverarbeitungssystems, einschließlich Kommunikation, wurden in LabWindows CVI in der Sprache C realisiert. Dabei sind mehrere Teilprozesse der Applikation entstanden [WGA13]. Allgemein wurden die Algorithmen der Signalverarbeitung unter Benutzung von Bibliotheken (DLL-*Dynamic Link Library*) umgesetzt. Die integrierte Festplatte des eingebetteten PXI-Controllers wird zur Abspeicherung der erfassten Daten (von den zwei analogen Eingängen) und von Ergebnissen (gefilterte und nicht gefilterte Masse) verwendet. In dem Benutzerprogramm auf dem Windows-PC werden die Messungen konfiguriert und die Ergebnisse angezeigt. Über die Ethernet-Schnittstelle erfolgt das Kopieren der Messwerte von der Festplatte des PXI-Controllers auf die interne Festplatte des Arbeitsrechners.

In der Promotion von Frau Baumgartl [Ba15] wurde u.a. dieses Vorläufersystem (Prototyp) näher dargestellt. Mit dem auf dem PXI-Controller dafür implementierten Informationsverarbeitungssystem ist die Latenz des geschlossenen Regelkreises 0,1 ms. Dabei ist die Ausführungszeit der Algorithmen (Regler und Filter) jitterbehaftet. Dieses Verhalten erfüllt die gestellten Anforderungen im Vorläufersystem, zeigt aber ein großes Potential zur Leistungssteigerung bei: Verringerung der Messunsicherheit, Erhöhung der Auflösung und der Reproduzierbarkeit, deutliche Verkürzung der Messzeit.

5.2 Exemplarische Umsetzung von ausgewählten Schritten der Prototypenentwicklung

Zum Nachweis der praktischen Anwendbarkeit des entwickelten Prozesses ZEfIRA war ein Ziel der vorliegenden Arbeit die exemplarische Entwicklung eines evolutionären Prototyps für Informationsverarbeitungsaufgaben in der Messtechnik.

Ausgangspunkt bildete die Analyse des Vorläufersystems, welche für die Modellbildung und die Weiterentwicklung teilweise adaptiert wurde. Der neue Prototyp sollte unterschiedliche Strategien der digitalen Signalverarbeitung unterstützen, flexibel auf Änderungen von Anforderungen reagieren und in das gesuchte, für das konkrete System geforderte Endprodukt münden können. Eine weitere Aufgabe war, die Untersuchungen des messtechnischen Teils zu unterstützen. In diesem Fall sollte das Informationsverarbeitungssystem auch zur Systemidentifikation verwendet werden. Außerdem wurden bei der Prototypenentwicklung die speziellen Anforderungen der Produktentwicklung, wie die Eichfähigkeits- und Zertifizierungsanforderungen und die Erfüllung der Metrologischen Sicherheit berücksichtigt.

Auf Grund der Komplexität hätten mit ZEfIRA nicht alle der hier interessierenden Aspekte vollständig untersucht werden können, wenn man sich nur auf ein abgeschlossenes Projekt bezogen hätte. Deshalb wurden ergänzende praktisch orientierte Arbeiten auch zu weiteren, nicht unmittelbar in den genannten Prototypen eingeflossenen Funktionen, durchgeführt.

Bei der exemplarischen Prototypenentwicklung wurde eine durchgängige Werkzeugunterstützung in der Entwicklungsumgebung LabVIEW mit Erweiterungen für Echtzeit- und FPGA-Plattformen verwendet. Nachstehend werden zusätzlich Möglichkeiten der Integration von anderen Modellierungs- und Entwicklungswerkzeugen in LabVIEW gezeigt. In diesem Fall wird eine effiziente Umsetzung aller Entwurfsetappen von der Anforderungsanalyse bis zur Implementierung entsprechend dem ZEfIRA Prozess erzielt. Alternativ wäre die Anwendung von MATLAB mit den Erweiterungen Simulink und Stateflow möglich [Mü12], wobei für das vorliegende Projekt LabVIEW als geeignet eingeschätzt wurde (s. Abschnitt 5.1).

5.2.1 Konzept und Werkzeugunterstützung zur Anforderungsanalyse

Im Hinblick auf eine Möglichkeit zur teilweise automatisierten und in ZEfIRA integrierten Anforderungsanalyse und -nachvollziehbarkeit wird im Folgenden ein exemplarisches Beispiel unter Verwendung des NI Requirements Gateway in der Entwicklungsumgebung LabVIEW dargestellt. Dieses basiert auf Prinzipien, die im Abschnitt 4.1 beschrieben wurden.

Die Abbildung 5.2 zeigt das Konzept der Umsetzung mit dazugehörigen Werkzeugen, das zur Untersuchung der speziellen Anforderungen der EMK-Waage für die Dissertation verwendet wurde. Darin ist das Zusammenwirken der voneinander abhängigen Entwicklungsphasen, beginnend mit der Anforderungsanalyse und deren Wechselwirkungen bis zum technischen Entwurf, zu erkennen.

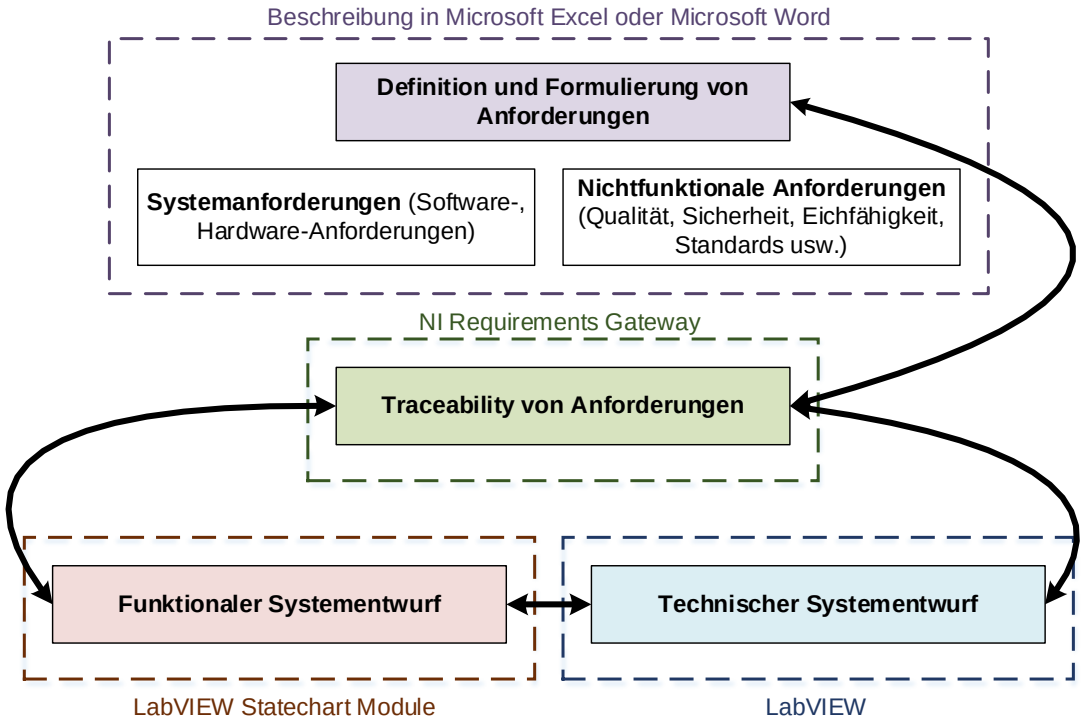


Abbildung 5.2: Konzept der Anforderungsanalyse in LabVIEW

Als erster Schritt werden die ermittelten Anforderungen definiert und formuliert. Deren Beschreibung kann z.B. in Microsoft Excel oder Word erfolgen. Zur Darstellung vieler Anforderungen unterschiedlicher Natur ist es sinnvoll eine logische Organisation bei der Beschreibung zu verwenden. In [Ch14] wurden mehrere Arbeitsblätter in Microsoft Excel erstellt und die Anforderungen in systembezogene und nichtfunktionale unterteilt. Für die Verfolgbarkeit (engl. *Traceability*) von Anforderungen über den gesamten Entwicklungsprozess wird anschließend das Werkzeug NI Requirements Gateway verwendet (Beispiele s. Anhang A.4). Dadurch werden Verbindungen zur modellbasierten Entwicklung in LabVIEW hergestellt und die Validierungsetappen unterstützt. Beim funktionalen Systementwurf ist es konzeptionell sinnvoll, mit Zustandsdiagrammen auf höherer Abstraktionsebene zur Darstellung der groben Systemarchitektur zu arbeiten. Mittels LabVIEW Statechart Module können die Beziehungen zwischen Systemteilen und dazugehörigen Anforderungen modelliert werden. Bei der Verfeinerung im funktionalen oder technischen Entwurf sind die entworfenen Modelle zu dokumentieren, wobei die Identifikationsbezeichnung jeder definierten Anforderung auch zu beschreiben ist.

Die schwerpunktmäßige Verwendung des NI Requirements Gateway bei der Anforderungsanalyse unterstützt Möglichkeiten zum Testen. Zuerst wird überprüft, ob alle Anforderungen dokumentiert beziehungsweise auch mittels Identifikationsbezeichnungen beschrieben sind. Weiterhin werden die Anforderungen mit Modellen (Systemteilen) verknüpft und bei der Überprüfung kann die Abdeckung der definierten Anforderungen im modellierten System untersucht werden.

Die Abbildung 5.3 präsentiert die Umsetzung des Konzeptes im NI Requirements Gateway, die im konkreten Projekt der EMK-Waage zum Einsatz kam. In diesem Zusammenhang unterstützt die erarbeitete und hier übersichtsmäßig beschriebene Variante eine teilautomatisierte Anforderungsanalyse und –nachvollziehbarkeit im Entwicklungsprozess.

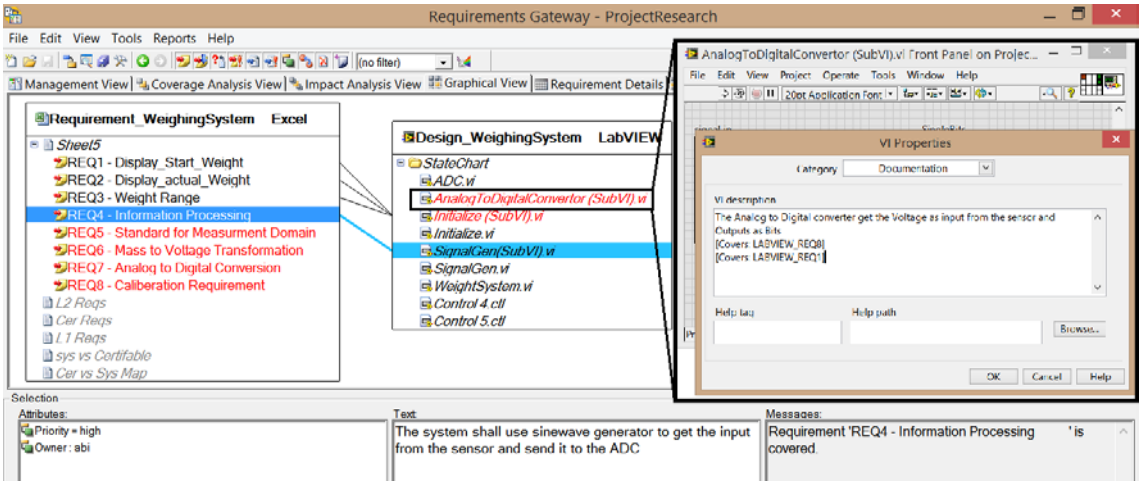


Abbildung 5.3: Anforderungsanalyse mittels NI Requirements Gateway

5.2.2 Funktionaler und technischer Systementwurf

Im folgenden Abschnitt wird die plattformunabhängige modellbasierte Entwicklung von ausgewählten Teilen des Informationsverarbeitungssystems dargestellt. Die Werkzeugunterstützung durch LabVIEW erlaubt den Entwurf von Systemen auf verschiedenen Abstraktionsebenen, ähnlich wie z.B. das für diese Entwicklungsetappe auch verwendete Entwurfswerkzeug MATLAB/Simulink [Mü12].

Die Abbildung 5.4 stellt die Komponenten der Informationsverarbeitungskette auf drei unterschiedlichen Abstraktionsebenen dar, die für diese Arbeit realisiert wurden. Die oberste Ebene (hier Ebene 3) enthält die Modelle des Messsystems und des Informationsverarbeitungssystems. Dabei wurde auf dieser Ebene die erste Partitionierung durchgeführt, die die Komponenten der EMK-Waage von Teilen des Informationsverarbeitungssystems trennt. Der Teil EMK-Waage steht für die gesamte Mechanik und die elektronischen und elektrischen Baugruppen. Diese werden zur Realisierung der elektromagnetischen Kraftkompensation auf Grund des Lageindikator-Analogsignals, des analogen Stellsignals und eines analogen Messsignals, welches proportional zum tatsächlichen

Spulenstrom ist, benötigt. Das Informationsverarbeitungssystem übernimmt alle Steuerungsfunktionen für das Messsystem, die Massebestimmung und die Weiterverarbeitung.

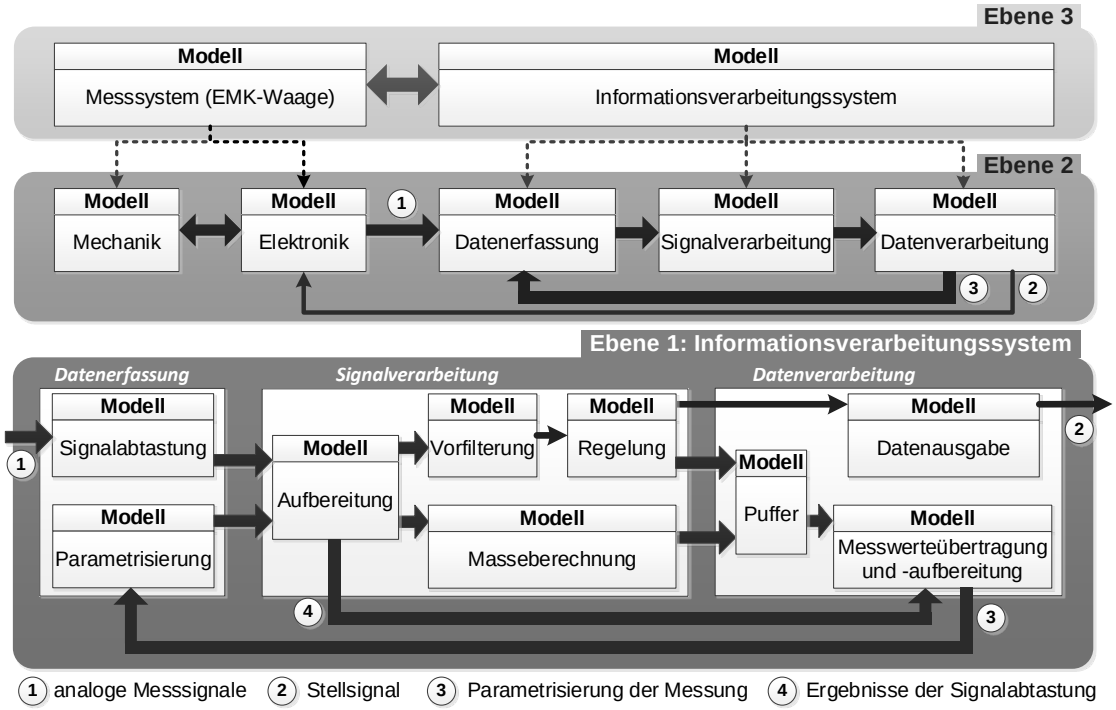


Abbildung 5.4: Systemkomponenten auf unterschiedlichen Abstraktionsebenen

Eine weitere Vertiefung erfolgt auf der Ebene 2. Das Modell des Messsystems wird in zwei Hauptmodelle unterteilt. In diesem Fall entstehen die Modelle der Mechanik und der Elektronik einschließlich der Elektromechanik mit ihren Schnittstellen. Diese Modelle wurden in der Dissertation von Baumgartl [Ba15] beschrieben und konnten hier aus dem Vorläufersystem teilweise übernommen werden. Wie bereits im Abschnitt 4.3.1 allgemein beschrieben, erwies sich auch für die konkreten Arbeiten an der Informationsverarbeitungslösung im Projekt eine weitere Entwicklung der Modelle als notwendig.

Im Weiteren werden die Komponenten des Informationsverarbeitungssystems genauer betrachtet. Es erfolgt eine Untergliederung in Datenerfassung, Signalverarbeitung und nachfolgende Datenverarbeitung. Dabei liefert die Datenerfassung die Digitalwerte der gemessenen Signale und eingestellten Parameter. Die Signalverarbeitung realisiert die Steuerung im engeren Sinne einschließlich der Masseberechnung. Die Datenverarbeitung dient zur Ausgabe des Stellsignals zum Messsystem und zur Aufbereitung der Massewerte.

Die auf der Ebene 2 entstandenen Modelle werden auf der Ebene 1 detaillierter beschrieben. Dabei werden die Funktionen und die Aufgaben jeder Teilkomponente bestimmt. Dieser Modellierung widmen sich die folgenden Abschnitte.

5.2.2.1 Entwurf des Messdatenerfassungssystems

Das Messdatenerfassungssystem ist ein wesentlicher Teil in der Informationsverarbeitungskette des für diese Arbeit entwickelten neuen Prototyps. Unter dem Messdatenerfassungssystem wird eine Funktionseinheit verstanden, die die Komponenten zur Signalabtastung mehrerer analoger Eingänge und die Komponenten zur Parametrisierung der Signalverarbeitung enthält. Bei Letzterem müssen die Daten beziehungsweise die Befehle vom Nutzer (Entwickler) erfasst und interpretiert werden.

Außer den Hauptaufgaben zur Signalabtastung und zur Parametrisierung wird dieses System um eine zusätzliche Funktion zur Systemidentifikation erweitert. Bei der Untersuchung der Dynamik und des Verhaltens der Einzelkomponenten der Messkette sowie bei der Analyse des Streckenverhaltens im messtechnischen System werden Möglichkeiten zur Systemidentifikation gebraucht. Dafür wurde ein zusätzlicher analoger Eingang (Reserve) bei der Signalabtastung vorgesehen. Es wurden Funktionen zur Signalgenerierung (in der Komponente „Parametrisierung“ von Abb. 5.4) und Signalanpassung (in der Komponente „Signalabtastung“ von Abb. 5.4) realisiert. Die Abbildung 5.5 enthält die schematische Darstellung und das LabVIEW-Modell des Messdatenerfassungssystems, das im Prototyp unter anderem zum Testen und zur Systemidentifikation verwendet wird.

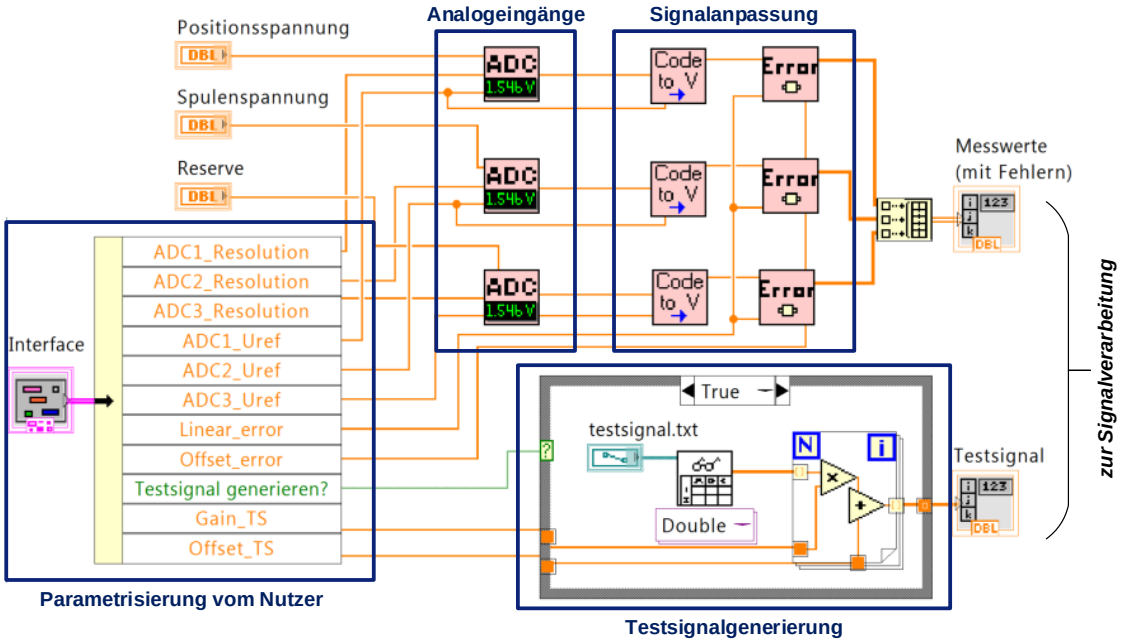


Abbildung 5.5: Modellierung des Messdatenerfassungssystems

Die folgenden realisierten Modelle sind zu erkennen:

1. *Drei analoge Eingänge für die Abtastung der benötigten Messsignale (Positionsspannung, Spulenstrom und Reserve-Kanal) mit dazugehörigen Modellen der Analog-Digital-Umwandlung (ADCs)*

Die Modellierung von Analog-Digital-Umsetzern wurde im Abschnitt 4.3.1 erläutert. Die hier dargestellte LabVIEW-Realisierung enthält Erweiterungen bezüglich der Parametrisierung. In diesem Fall konnte eine Untersuchung der Umsetzer bezüglich unterschiedlicher Auflösungen, Abtastraten und Referenzspannungen durchgeführt werden.

2. *Modelle der Signalanpassung*

Die abgetasteten Signale werden von Analog-Digital-Umsetzern in Form eines Codes dargestellt und zu entsprechenden Spannungswerten umgerechnet. Diese Modelle enthalten Funktionen zur Fehlerberechnung (s. auch Abschnitt 4.3.3). Als Ergebnis werden die Messwerte mit ihrem absoluten Fehler für die nachfolgende Signalverarbeitung ausgegeben.

3. Modell des Interfaces

Dieses Modell dient zur Parametrisierung des Messdatenverarbeitungssystems. Dabei entsteht die Möglichkeit, die verwendeten analogen Eingänge zu parametrisieren und die Funktion der Testsignalgenerierung zu aktivieren.

4. Modell der Testsignalgenerierung

Bei der Aktivierung durch den Nutzer (Entwickler) können unterschiedliche generierte Funktionen (z.B. Sinus, Chirp-Signal, Pseudorauschen) anstelle der Stellgröße des entsprechenden Reglerausganges geladen werden. In dieser Funktionsart zur Systemidentifikation werden die Ergebnisse der Signalabtastung in der Datenverarbeitung zur späteren Auswertung abgespeichert. Weiterhin werden die Komponenten der Signalverarbeitung (Vorfilterung, Regelung und Masseberechnung) und die Gewichtsausgabe abgeschaltet. Alle genannten Komponenten sind in der Abbildung 5.4 angeordnet.

Das dargestellte Messdatenerfassungssystem mit der erweiterten Funktionalität der Signalgenerierung wird in diesem Zusammenhang nur im Prototyp verwendet. Dabei spielen die Systemidentifikation und die zusätzlichen Testmöglichkeiten eine wichtige Rolle. Im Abschnitt 5.2.3.1 wird die plattformabhängige Entwicklung dieses Systems beschrieben. Außerdem werden die Partitionierung und die Aufgaben der entstandenen Partitionen detaillierter dargestellt.

Bei der Entwicklung einer produzierbaren Lösung wird eine untersuchte Konfiguration realisiert, die auf Basis des W-Prozesses „Entwicklung eines Prototyps“ eine für die konkrete Anwendung optimierte Lösung zur Datenerfassung verwendet. Das bedeutet, dass nicht alle Funktionen (z.B. Testsignalgenerierung) enthalten sein werden.

5.2.2.2 Entwurf des Signalverarbeitungssystems

Als Weiteres beschäftigt sich die vorliegende Arbeit mit der Entwicklung des digitalen Signalverarbeitungssystems. Dieses wurde modellbasiert entworfen. Die Hauptaufgabe

ist die Entwicklung eines Regelungssystems auf Basis der erstellten Modelle. Die durchgängige Entwurfsmethodik nach ZEfIRA erlaubt es, im Prototyp entwickelte Regelungskonzepte für eine breitere Klasse von Wägesystemen zu unterstützen, sofern ein detailliertes Systemmodell des zu regelnden mechatronischen Systems erstellt wird. Die durchgeführten Simulationen ermöglichen dabei Änderungen mit Blick auf eine optimale Lösung für die konkrete Waage.

Das Signalverarbeitungssystem ist schematisch in der Abbildung 5.6 gezeigt.

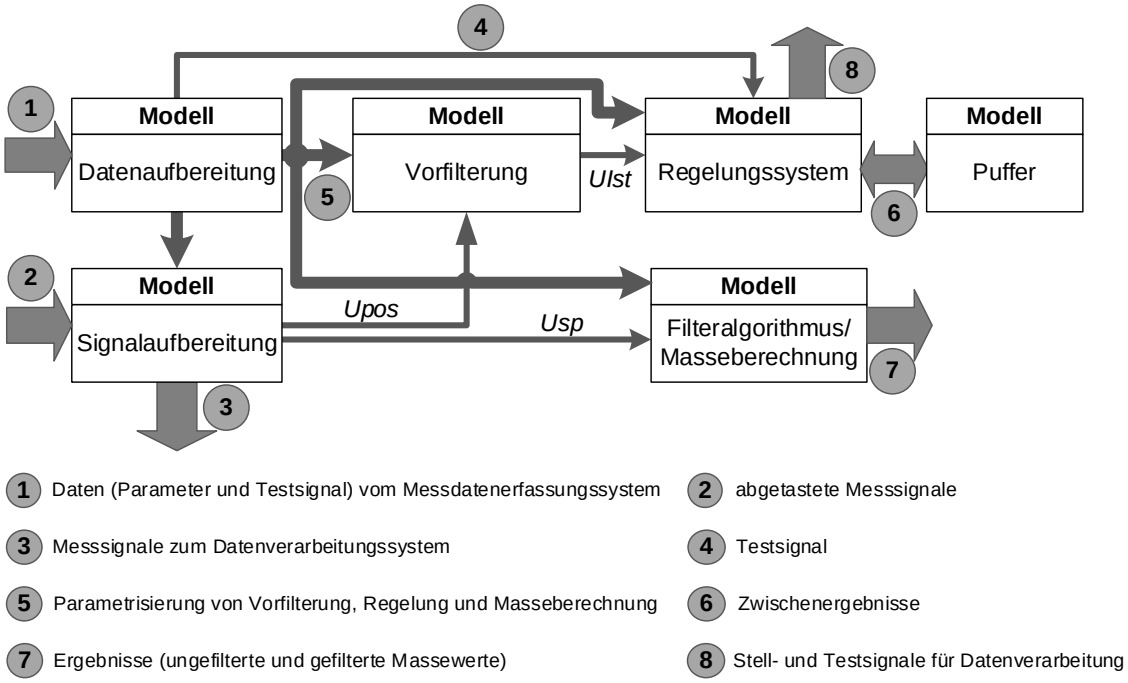


Abbildung 5.6: Komponenten des Signalverarbeitungssystems

Die Komponente der Datenaufbereitung erhält die Daten vom Messdatenerfassungssystem. Diese sind die Werte zur Parametrisierung von Algorithmen der Signalverarbeitung, sowie des generierten Testsignals, das im Prototyp zur Systemidentifikation und/oder zum Testen verwendet wird (1). Die Signalaufbereitung ist eine Bearbeitungsfunktion, die die abgetasteten Messsignale (Positionsspannung, Spulenspannung und Reserve (2)) für die Signalverarbeitung erhält und diese zur Datenübertragung (s. Datenverarbeitungssystem) vorbereitet. Die weiteren Funktionsgruppen (Vorfilterung, Regelungssystem, Filteralgorithmus/Masseberechnung) benötigen an dieser Stelle eine besondere Betrachtung, weil diese die Hauptkomponenten der neuentwickelten Signalverarbeitung sind. Das Konzept beziehungsweise die Algorithmen der Filterung und der Masseberechnung wurden aus dem Vorläufersystem übernommen und sind im Abschnitt 5.2.3.4 detaillierter dargestellt.

Die **Vorfilterung** dient zur Verbesserung der Mess- und Regelgüte. Dabei werden die abgetasteten Messsignale aufbereitet.

Diese Funktion wurde im Vorläufersystem zur Rauschunterdrückung verwendet und war in Form eines analogen Tiefpassfilters realisiert. Bei der Entwicklung des neuen Prototyps wurde mit dieser Komponente eine Erhöhung der Auflösung der Sensorsignale (abgetastete Messwerte: *Upos*) realisiert. Zur Untersuchung dieser Funktion wurden Messungen zur Systemidentifikation durchgeführt [AKRW14]. Für den funktionalen Systementwurf wurden die gemessenen Kennlinien zu einer modellbasierten Untersuchung der Vorfilterung in LabVIEW verwendet, welche das abgebildete Verhalten der analogen Filterung nutzte. Ein darauf aufbauender detaillierter Entwurf wurde beim technischen Systementwurf durchgeführt. Auf Basis einer FFT-Analyse konnten die signifikanten Frequenzen im Amplituden-Frequenzgang ermittelt werden. Danach wurden klassische IIR- und FIR-Tiefpassfilter entwickelt, die eine Dämpfung hoher Frequenzen mit einer Grenzfrequenz von 10 kHz ermöglichen und in dieser Applikation zum Einsatz kommen können. Die Simulationsergebnisse zeigten, dass so eine mit dem Analogfilter vergleichbare Störunterdrückung möglich ist. Außerdem konnte ein FIR-Filter als gleitender Mittelwertbildner in der Kombination mit einer „Downsampling“-Funktion zur Erhöhung der Anzahl der effektiv nutzbaren Bits bzw. der Auflösung der Messsignale verwendet werden. Die Details der Umsetzung werden im Abschnitt 5.2.3.2 beschrieben. Es werden Partitionierungsmöglichkeiten gezeigt, die einen optimalen Kompromiss zwischen Rechenaufwand, Leistung und Ressourcenverbrauch erzielen.

Der **Regelalgorithmus** ist der aufwendigste Teil des Signalverarbeitungssystems. Es sollte im Sinne eines „Prototyps“ die Möglichkeit unterstützt werden, die verschiedenen Regelungskonzepte zu untersuchen, um die Leistungs- und Genauigkeitsanforderungen im Vergleich zum Vorläufersystem zu erfüllen

Der modellbasierte Entwurf eines Regelungssystems benötigt ein detailliertes Systemmodell. Dabei sollen aus der Sicht der Regelungsentwicklung ein Streckenmodell, ein Modell zusätzlicher Informationsverarbeitungskomponenten (hier auch Umgebungsmodell genannt) und ein Modell des Reglers eng zusammen betrachtet werden.

Das Streckenmodell enthält die Partitionen für den mechanischen und elektronischen Aufbau der EMK-Waage (in Abb. 5.4). Die Kenntnisse des mechanischen Systems sind entscheidend sowohl für die dynamischen, als auch für die statischen Messfehler. Die starke Verflechtung von Mechanik, Elektronik und Signalverarbeitung erschwert die Arbeiten zur Verbesserung der messtechnischen Eigenschaften erheblich. Die Zusammenhänge zwischen Ursache und Wirkung in diesen rückgekoppelten komplexen Systemen sind nur schwer trennbar. Das Streckenmodell besteht aus dem linearen Modell der EMK-Waage (mechanisches Modell) und den Schnittstellen zur Umwandlung der Ein- und Ausgabegrößen (Spannungen) in die mechanischen Größen. Wie im Abschnitt 4.3.1.1 beschrieben, wurden dabei die Kenntnisse aus dem Vorläufersystem übernommen und für die vorliegende Dissertation als Vorlage der Modellierung des Systemmodells in LabVIEW verwendet.

Das Umgebungsmodell enthält ein abstraktes Modell der Ablaufsteuerung (zur Parametrisierung und Steuerung der Regelung), Modelle des Analog-Digital- und des Digital-Analog-Umsetzers einschließlich Vorfilterung und einer Anzeige zur Überwachung der Zwischenergebnisse. In diesem Fall umfasst das Umgebungsmodell die oben beschriebenen Komponenten des Messdatenerfassungssystems und der Datenverarbeitung.

Die Abbildung 5.7 stellt das Systemmodell der Regelung schematisch dar.

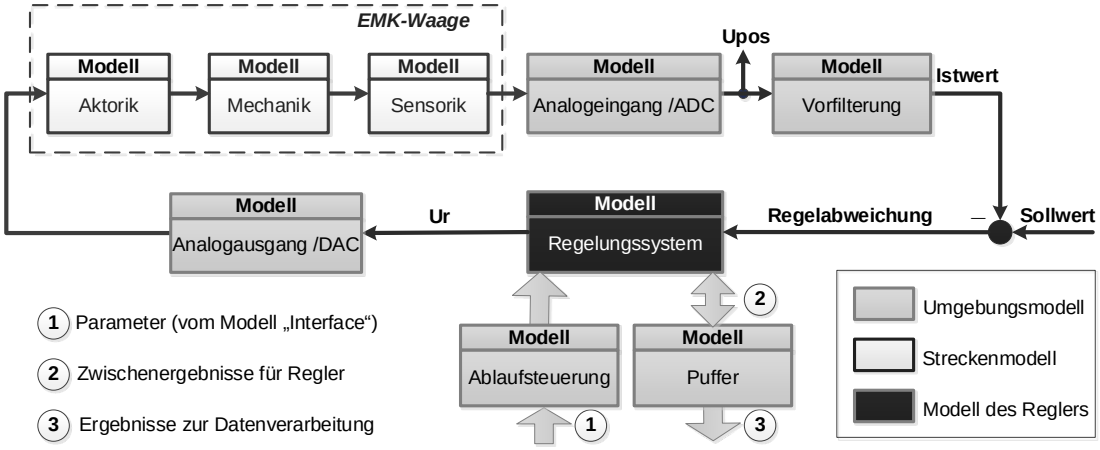


Abbildung 5.7: Systemmodell des Regelsystems

Der Eingangswert des Reglers (Regelabweichung) wird aus dem Unterschied der Führungsgröße (Sollwert) mit der gemessenen und zurückgeführten Regelgröße (Istwert) ermittelt. Dabei ist der Sollwert entsprechend dem EMK-Prinzip Null (Steuerung des Hebels in die Nulllage) und der Istwert ist die Positionsspannung. Diese Spannung wird vom Sensor im Messdatenerfassungssystem abgetastet und für die Signalverarbeitung aufbereitet. Die Tiefpass-Vorfilterung liefert den behandelten Wert der Positionsspannung als Istwert zum Regler.

Die Realisierung der in der Abbildung 5.7 dargestellten Komponenten des Systemmodells mit der Entwicklungsumgebung LabVIEW benötigte neben den oben beschriebenen und realisierten Modellen des Messdatenerfassungssystems und der Signalverarbeitung auch die Umsetzung des Streckenmodells.

Zur Modellierung des mechanischen Systems wurde die Dynamik der Waage durch Bewegungsdifferenzialgleichungen beschrieben (s. Abschnitt 4.3.1.1). Die Schnittstellen der Umwandlung der physikalischen Größen (elektronische und elektrische Funktionsgruppen) wurden durch identifizierte Kennlinien dargestellt. Dabei wurden die wegabhängige Kraftkonstante (Kraft-Strom-Kennlinie), der Positionssensor (Spannung-Weg-Kennlinie) und die U/I-Umwandlung (Spannung-Strom- Kennlinie) modelltechnisch berücksichtigt. Zur System- beziehungsweise Komponentenidentifikation wurden das Messdatenerfassungssystem und die nachfolgende Datenverarbeitung verwendet.

Das Modell des Regelungssystems enthält neben dem mathematischen Modell des Reglers (Übertragungsfunktion) die zusätzlichen Schnittstellen zur Umwandlung der Messgrößen (z.B. Umwandlung der Kraft in den Eingangswert des DA-Umsetzers).

Die Abbildung 5.8 zeigt die LabVIEW-Realisierung der beschriebenen Komponenten des Systemmodells.

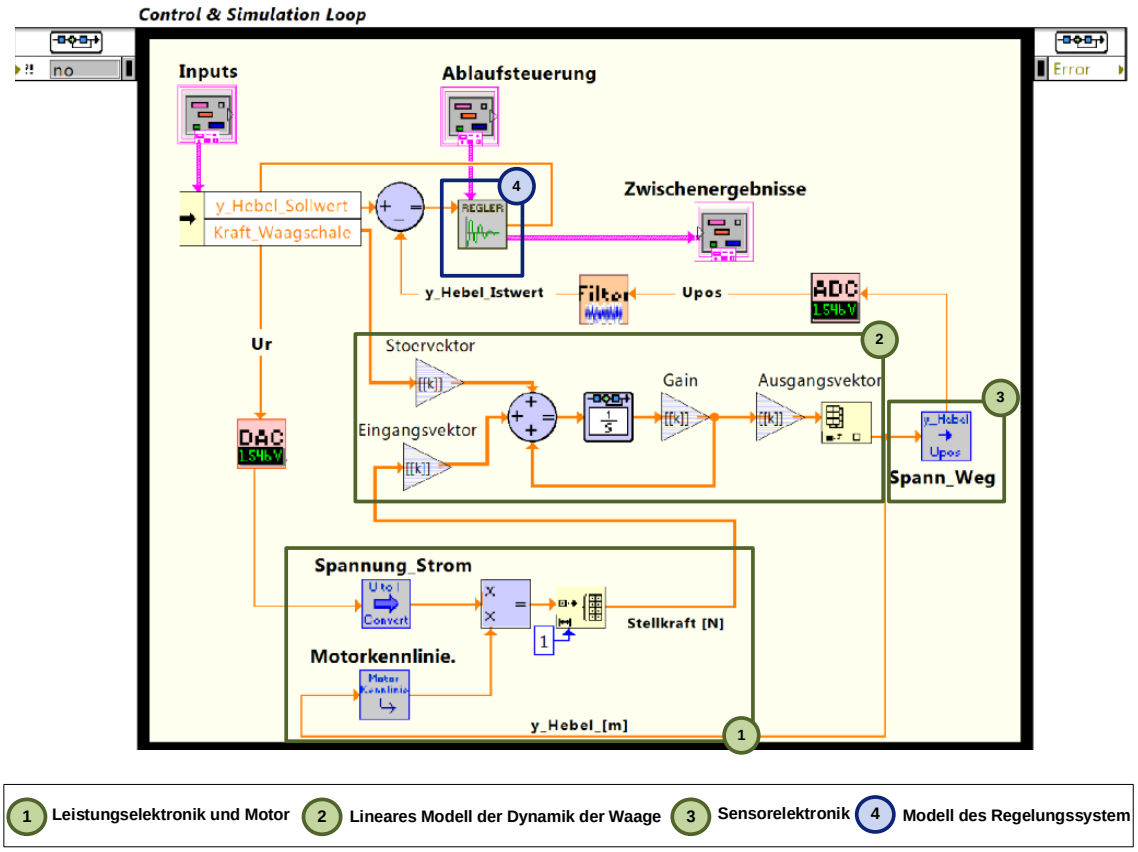


Abbildung 5.8: Simulationsmodell des modellierten Systems

Mit diesem Modell wurde das Verhalten der EMK-Waage simuliert. Dabei konnten unterschiedliche klassische Regelungsalgorithmen untersucht werden. Die Implementierung des Regelungssystems wird im Abschnitt 5.2.3.3 näher erläutert.

5.2.3 Verfeinerter technischer Entwurf und Synthese zur Realisierung der Signalverarbeitungsaufgaben

Dieser Abschnitt stellt den verfeinerten technischen Entwurf des Informationsverarbeitungssystems der EMK-Waage dar. Dabei werden die Schritte des plattformabhängigen modellbasierten Entwurfes einschließlich verschiedener Partitionierungsmöglichkeiten sowie die nachfolgende Synthese zur Realisierung der oben genannten Signalverarbeitungsaufgaben beschrieben. Das beinhaltet die Etappen der Partitionierung EBS-EGS (4), des verfeinerten technischen Entwurfs (5), der Partitionierung des EGS (6) einschließlich

Entwurf/Synthese der SW-, HW- und FPGA-Module im W-Modell „Prototyp“ (entsprechend der Abb. 3.2, im Abschnitt 3.1).

Zuerst wird das Gesamtsystem in die Komponenten des Einbettenden und Eingebetteten Systems geteilt (Abb. 5.9). Zum Einbettenden System gehört die EMK-Wägezelle einschließlich der Elektronik und Elektromechanik im Wägesystem. Das Eingebettete System stellt das Informationsverarbeitungssystem dar. Außerdem wurde im Prototyp eine Mensch-Maschine-Schnittstelle am Gesamtsystem gebraucht. Ein zusätzlicher PC mit dem Betriebssystem Windows realisiert diese.

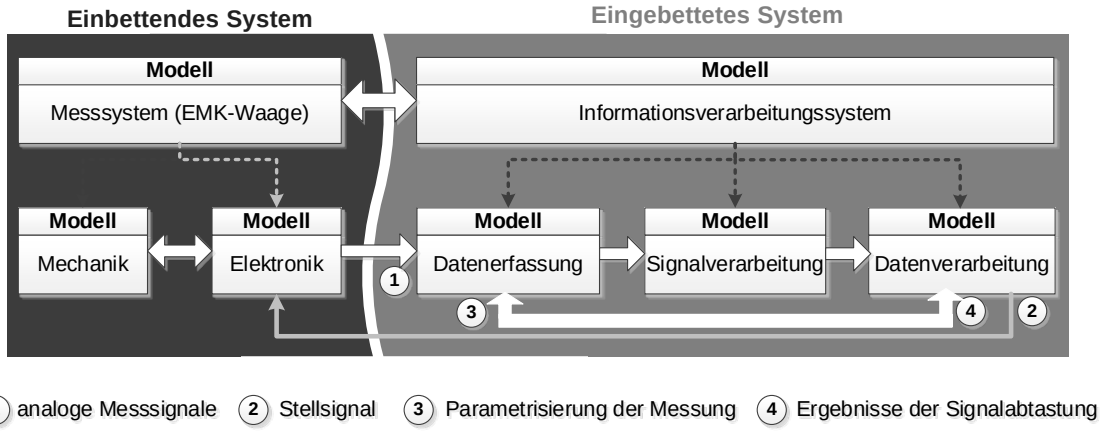


Abbildung 5.9: Partitionierung des Gesamtsystems

Bei der Entwicklung des Prototyps zur Messdatenerfassung und –bearbeitung wurden industrielle Baugruppen verwendet. Das sollte ein stabiles und kurzfristig verfügbares Hardware-System mit einer entsprechenden modellbasierten Entwicklungsumgebung gewährleisten. In dem beschriebenen Konzept ZEfIRA (s. Kapitel 3) spielen leistungsfähige und kostenintensive FPGA-basierte Eingebettete Systeme bei der prototypischen Entwicklung eine besondere Rolle. Diese sind eine verbreitete Lösung für prozessgekoppelte Signalverarbeitungsaufgaben mit besonders geringer Latenz. Sie ermöglichen vor allem Flexibilität, modellbasierten Entwurf und hohe Leistungsfähigkeit zum flexiblen Testen verschiedener Varianten von Algorithmen und Rechnerarchitekturen (Kombinationen von Hardwarestrukturen und Softwarelösungen). [Mü12][ZKGA13]

Die Implementierungsplattform für die Informationsverarbeitungsaufgaben in dem Vorläufer-EMK-Wägesystem war ein PXI-basiertes Echtzeitsystem von National Instruments ohne FPGA-Baugruppen. Hierzu wurden ein PXIe-Controller und ein AD/DA-Modul für die Signalverarbeitung verwendet (s. Abschnitt 5.1). Darauf basierend wurde bei der Entwicklung des neuen Prototyps eine erweiterte Plattform ausgesucht, die zusätzlich Module anbietet, welche sowohl Schnittstellen zur analogen Ein- und Ausgabe als auch ein rekonfigurierbares FPGA beinhalten. Für diese Funktionalität wurde ein

Multifunktions-RIO-Modul ausgewählt, das im PXI-System platziert werden kann und mit dem PXI-Controller standardmäßig über den PCI-Bus kommuniziert.

In dieser Arbeit wurde das FPGA-Modul NI PXI-7854R gewählt, das ein leistungsfähiges FPGA Virtex-5-LX110 einschließlich Onboard-Speicher (512 kB) und direkt angebundene analoge Ein- und Ausgänge besitzt. Dabei stehen acht voneinander unabhängige Analogeingänge (16 Bit Auflösung), acht Analogausgänge (16 Bit Auflösung), 96 bidirektionale Digitalanschlüsse und drei Kanäle zum direkten Speicherzugriff (engl. *Direct Memory Access* – DMA) zur Verfügung [RIO14]. Die letztgenannten Komponenten ermöglichen eine Hochgeschwindigkeitsdatenübertragung zwischen dem PXI-Controller und dem FPGA-Modul.

Die Kombination aus einem eingebetteten PXI-Controller, dem FPGA-Chip (eventuell mit dem für den Prototyp entwickelten Softcore-Prozessor) und zusätzlichen analogen und digitalen Funktionseinheiten ergibt die spezifizierte Hardware-Gesamtfunktionalität. Insbesondere ermöglicht sie iterative Repartitionierungen auf unterschiedlichen Abstraktionsebenen.

Der verfeinerte technische Entwurf nutzt die plattformspezifische Modellierung, die eine abstrakte Repräsentation von Plattformressourcen und ihren Schnittstellen auf Modellebene darstellt. Dabei werden die Modelle der plattformunabhängigen Entwicklung (s. Abschnitt 5.2.2) in der Entwicklungsumgebung LabVIEW (mit Erweiterungen für Echtzeitcontroller und FPGA) verfeinert. Die definierten Komponenten der Informationsverarbeitung mit ihren Datenflüssen werden im Folgenden mit unterschiedlichen Partitionierungsmöglichkeiten beschrieben. Entsprechend dem Prozess ZEfIRA wird das Eingebettete System weiterhin in die nachstehenden Teile partitioniert: zusätzliche analoge und digitale Hardware, FPGA-Logik, im FPGA enthaltener Softcore-Prozessor mit Programm und Controller mit Controller-Software.

Die Abbildung 5.10 stellt die Informationsverarbeitungskette mit unterschiedlichen Partitionierungsmöglichkeiten dar, die in dieser Arbeit untersucht wurden.

Entsprechend der gestellten Anforderungen werden in allen Partitionierungsvarianten die zeitkritischen Aufgaben (z.B. Signalabtastung und Regelung) mit der FPGA-basierten Hardware realisiert. Die Messwertaufbereitung verwendet andererseits den PXI-Controller als Plattform. Es wurden darauf basierend unterschiedliche Varianten untersucht:

1. ***der erweiterte Prototyp*** dient zur maximalen Unterstützung des Entwicklungsprozesses und kann sowohl für die allgemeinen Informationsverarbeitungsaufgaben in der EMK-Waage als auch die hochauflösende und latenzarme Signalerfassung und -ausgabe während der experimentellen Prozessanalyse in Echtzeit verwendet werden;

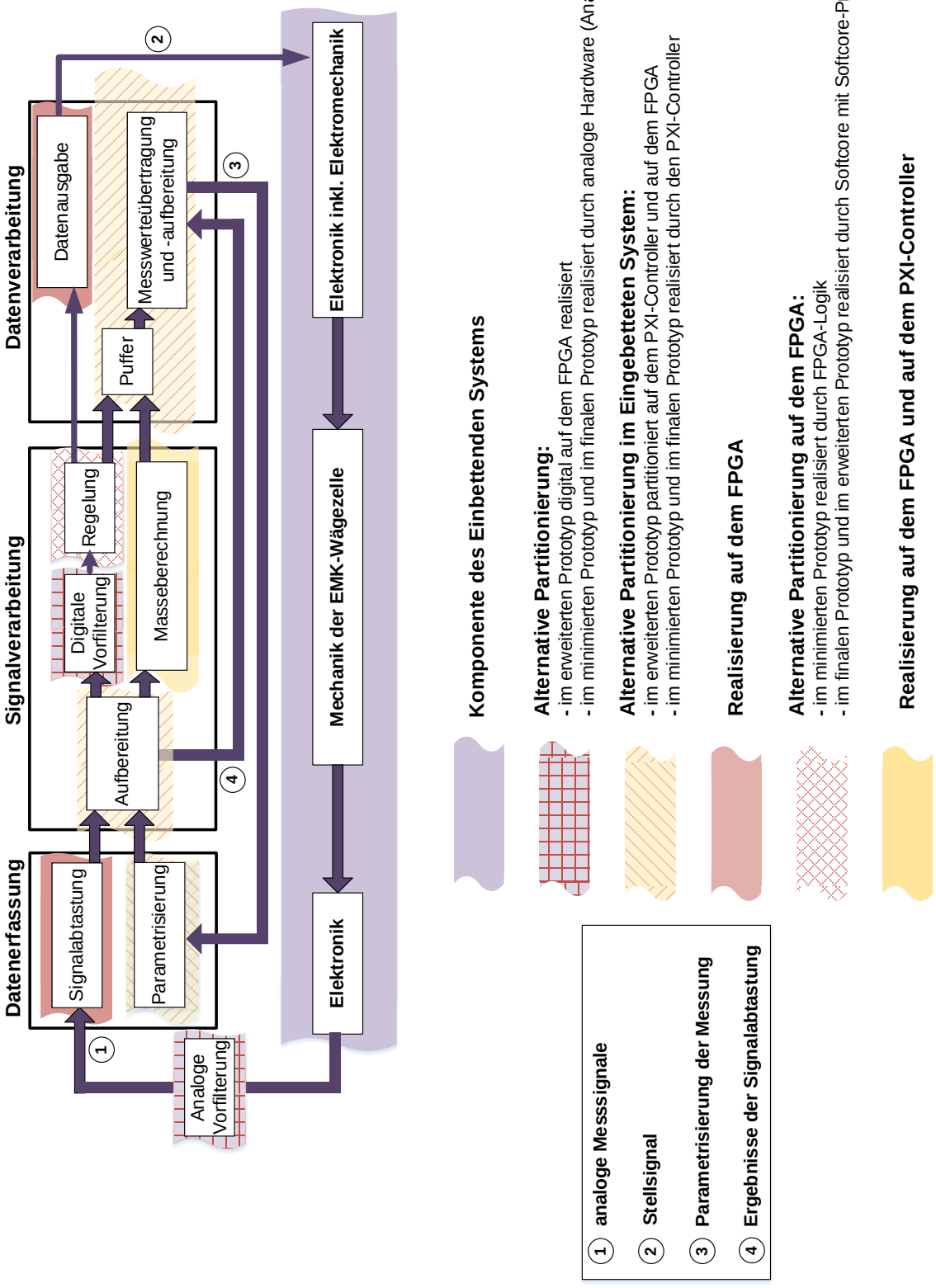


Abbildung 5.10: Partitionierungsmöglichkeiten im Informationsverarbeitungssystem

2. ***der finale Prototyp*** ermöglicht die Weiterführung der Entwicklung des Einbettenden Systems mit relativ fixierter Informationsverarbeitung und nähert sich einer produzierbaren Lösung an;
3. ***der minimierte Prototyp*** dient als direkte Umsetzung des Vorläufersystems auf die neue Implementierungsplattform, wobei aber Parameteränderungen und Leistungsverbesserungen möglich sind.

Durch die anschließenden Schritte der Synthese wurden die entworfenen und bereits beschriebenen Funktionen des Informationsverarbeitungssystems für die betrachtete Applikation umgesetzt. Dabei stellt die Entwicklung des Softcore-Programms in den entsprechenden Varianten ein spezielles Problem dar. Der Softcore-Prozessor inklusive eines Codekonverters und einer Testumgebung wurde speziell für diese Anwendung entwickelt, wie im Abschnitt 4.4.1.3 beschrieben, und in den durchgeführten Untersuchungen verwendet.

5.2.3.1 Messdatenerfassungssystem und Datenausgabe

Beim verfeinerten und plattformabhängigen Entwurf des Messdatenerfassungssystems wurden unterschiedliche Partitionierungen untersucht. Dabei wurden die Strategien des erweiterten Prototyps, des finalen Prototyps und des minimierten Prototyps unterstützt. Diese unterscheiden sich in der Parametrisierung und der optionalen Testsignalgenerierung. In den Fällen der minimierten und der finalen Realisierungslösung wurden die Funktionen zur Systemidentifikation und zum Testen (mit unterschiedlichen Testsignalen) nicht integriert. Die Komponente „Parametrisierung“ wird in beiden Strategien auf dem PXI-Controller realisiert und enthält die Daten zur Steuerung von Abläufen auf dem FPGA und zur Parametrisierung der Gesamtmessung. Im Weiteren wird die alternative Partitionierung im erweiterten Prototyp detaillierter dargestellt.

Die Verwendung eines FPGA und die Erfüllung von Anforderungen an die latenzarme Signalerfassung und Datenausgabe sind durch die Anwendung des oben beschriebenen Moduls NI PXI-7854R in allen genannten Strategien realisiert. Die charakteristischen Eigenschaften der eingesetzten Analog-Digital- beziehungsweise Digital-Analog-Schnittstellen werden an dieser Stelle näher erläutert.

Analoge Eingänge des NI PXI-7854R:

Zur Digitalisierung von Messwerten werden die AD-Umsetzer verwendet, die auf dem Modul zur Verfügung stehen.

Diese sind Umsetzer mit 16 Bit Auflösung nach dem Prinzip der sukzessiven Approximation mit bis zu 750 kHz Abtastrate. Der Eingangsspannungsbereich ist ±10V und kann bei diesem Modul nicht geändert werden. Dabei ergibt sich für 1 LSB (Least Signifikant Bit - niedrigste Bitposition) eine Spannung von ~305 µV.

Analoge Ausgänge des NI PXI-7854R:

Es stehen acht 16 Bit Analogausgänge (DAU nach einem parallelen Verfahren mit R2R-Netzwerken [BST10]) zur Verfügung. Die Spannungssignale können im Ausgangsbereich ± 10V mit einer Frequenz von bis zu 1 MHz ausgegeben werden. Im betrachteten Signalverarbeitungssystem wird ein DA-Umsetzer für die Ausgabe des Stellsignals *Ur* im geschlossenen Regelkreis verwendet. Dabei bestimmt die Auflösung des Umsetzers die Auflösung des gesamten Signalverarbeitungssystems. Außerdem besitzen diese DA-Umsetzer eine hohe Linearität sowie einen geringen Offset und eine geringe Drift. Dadurch können sie für die speziellen Zwecke der Messtechnik zum Einsatz kommen (s. auch [ZKGA13]).

Die Abbildung 5.11 zeigt eine grobe Darstellung des gesamten Messdatenerfassungssystems.

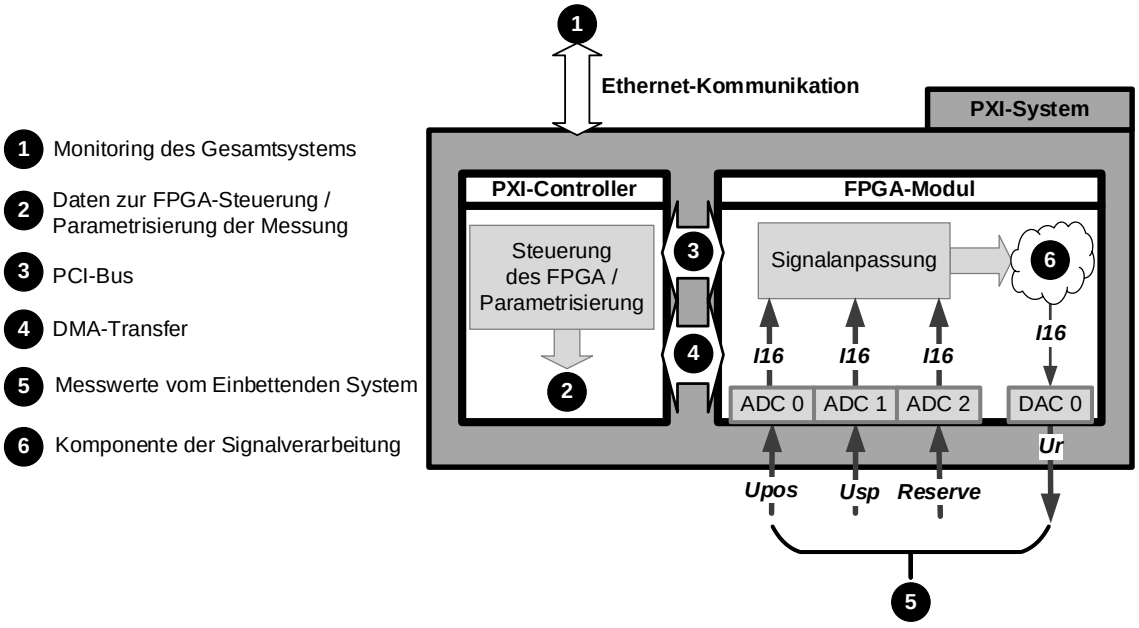


Abbildung 5.11: Übersicht des gesamten Messdatenerfassungssystems

Die im Abschnitt 5.2.2.1 beschriebenen Komponenten (Signalabtastung und –anpassung, Parametrisierung einschließlich Testsignalgenerierung) wurden auf den zwei Plattformen implementiert. Wie oben gesagt wurde, ist die Komponente „Signalabtastung“ mit dem FPGA-Modul umgesetzt worden. Die analogen Messsignale von der EMK-Waage (Positionsspannung, Spulenspannung und Reserve) werden über die analogen Eingänge des Moduls direkt zur Signalanpassung auf dem FPGA-Chip transportiert.

Der PXI-Controller wurde hauptsächlich für die allgemeinen Datenverwaltungsaufgaben verwendet. Hierzu wurden auf dem Controller die Aufgaben zur Steuerung und Berechnungen von Nebenfunktionen, zur Kommunikation mit Teilsystemen, zur Konfiguration des FPGA-Moduls sowie zur Abspeicherung von Datenmengen realisiert. Außerdem

wurde die Funktion der Testdatengenerierung im erweiterten Prototyp umgesetzt. In diesem Fall wird das gewünschte Signal (Sinus, Chirp-Signal oder Pseudorauschen) auf dem Controller generiert oder aus einer Datei gelesen und in Form eines Datenpaketes über FIFO-DMA zur Signalverarbeitung auf dem FPGA transportiert [AKRW14].

Im erweiterten Prototyp wird eine externe Kommunikation des PXI-Systems mit dem Arbeitsrechner benötigt. Dazu wird die Ethernet-Kommunikation zum Arbeitsrechner verwendet. Dieser realisiert die Abspeicherung von größeren Datenmengen, die für die Analyse des Messsystems oder zur Änderung von Aufgaben auf dem PXI-System notwendig sind.

Zur Umsetzung von Funktionen des Messdatenerfassungssystems und der Datenausgabe dient die Entwicklungsumgebung LabVIEW mit den Erweiterungen Real Time (Implementierung auf dem PXI-Controller) und FPGA (Implementierung auf dem FPGA-Modul). Dabei stehen alle benötigen Komponenten zur Verfügung. Bei der Konvertierung von Messwerten (in der Signalanpassung) wurde die speziell dafür entwickelte Bibliothek verwendet (s. Abschnitt 4.4.1.2).

5.2.3.2 Vorfilter

Die Abbildung 5.10 zeigt die alternativen Partitionierungsmöglichkeiten bei der Realisierung der Vorfilterfunktion. Bei dem minimierten und dem finalen Prototyp wurde diese Funktion aus dem Vorläufersystem vollständig übernommen. In diesem Fall ist der Vorfilter mit Analogelektronik aufgebaut und gehört deswegen zur Partition des Einbettenden Systems.

In diesem Abschnitt wird die Realisierung des Vorfilters als Teil des Eingebetteten Systems beschrieben. Bei dieser Partitionierung wurde die Vorfilterfunktion als digitaler Tiefpassfilter auf dem FPGA implementiert. Der grundsätzliche Vorteil durch die Realisierung im Eingebetteten System ist, dass die verschiedenen Filteralgorithmen flexibel untersucht werden konnten. Da es mit der FPGA-basierten Signalerfassung möglich ist, hohe Abtastraten zu erreichen, wurden der Einsatz und die Unterschiede von digitalen klassischen FIR und IIR Tiefpassfiltern in den folgenden ausgewählten Varianten untersucht.

Variante 1: FIR-Filter zur Auflösungserhöhung der Positionsspannung *Upos*

Diese Variante enthält die „Downsampling"-Funktion in Kombination mit einem Mittelwertbildner. In [Pf96] wurde sie zur Verringerung der Rauschanteile (unter der Annahme, dass weißes Rauschen vorliegt) und zur Erhöhung der Auflösung vorgeschlagen. Dabei kann die Standardabweichung des Rauschanteils um den Faktor *N* verringert werden, wobei „*N*" – die Anzahl von zusammenhängenden Messwerten ist. Dieses trägt zur Erhöhung der Anzahl von effektiv nutzbaren Bits bei. Die klassische Mittelwertbildung durch

Summation aufeinander folgender Messwerte und Division durch deren Anzahl ist eine Realisierungsmöglichkeit eines FIR-Filters.

In der vorliegender Arbeit wurde die Formel (5.1) mit der FPGA-Logik umgesetzt. Dabei wurden eine Addition und eine Multiplikation aus den entworfenen Floating-Point-Bibliotheken benutzt (s. Abschnitt 4.4.1.2).

$$\boldsymbol{Upos_{MWB}} = \frac{\mathbf{1}}{\boldsymbol{N}} \cdot \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{Upos_i} \qquad \textbf{(5.1)}$$

mit $Upos_i$ – auftretende Messwerte der Positionsspannung; $Upos_{MWB}$ – Positionsspannung nach der Mittelwertbildung; $\boldsymbol{N}$ – Anzahl von zusammenhängenden Messwerten.

Die Realisierung auf dem FPGA wurde im erweiterten Prototyp für unterschiedliche Parametrisierungen des Wertes „*N*“ untersucht. Durch die Parametrisierbarkeit vom Nutzer (beziehungsweise Entwickler) konnten verschiedene Kombinationen getestet werden, z.B. die Verringerung der Abtastrate von 280 kHz auf 10 kHz. In diesem Fall wird *N=28* eingegeben (Mittelwertbildung von 28 aufeinander folgenden Messwerten). Die maximale Abtastfrequenz der gesamten Signalverarbeitung einschließlich dieses Filters von 280 kHz ist durch die Komplexität aller gleichzeitig implementierten Signalverarbeitungsalgorithmen begrenzt (s. auch 5.2.3.3).

Variante 2: IIR-Filter

Eine weitere Variante stellt die Realisierung eines IIR-Filters zur Verbesserung der Mess- und Regelgüte dar. Die Umsetzung dieses IIR-Filters benötigt einen höheren Realisierungsaufwand als im Fall des FIR-Filters.

Als passendes Beispiel für die betrachtete Anwendung des IIR-Tiefpass-Filters wurde ein Elliptischer Filter in MATLAB untersucht. Es wurde ein geeigneter Filter auf Basis der maximalen Abtastfrequenz von 280 kHz und der Grenzfrequenz von 10 kHz gewählt. Dabei ergibt sich die Filterordnung *n*=28 (Abb. 5.12). Außerdem wurden die entsprechenden Koeffizienten des Filters berechnet.

Die Entwicklungsumgebung MATLAB/Simulink mit der Erweiterung FDATool [MW15] bietet die Möglichkeit der Datenflussdarstellung dieses Filters, die auch im Weiteren zur Implementierung auf dem FPGA verwendet wurde. Im Anhang A.1 ist die gekürzte Datenflussdarstellung des Elliptischen Filters 28.Ordnung gegeben. Die komplexe Struktur des Filters enthält mehrere Stufen (Sektionen), die zur Berechnung der Messgröße (hier für *Upos*) gebraucht werden.

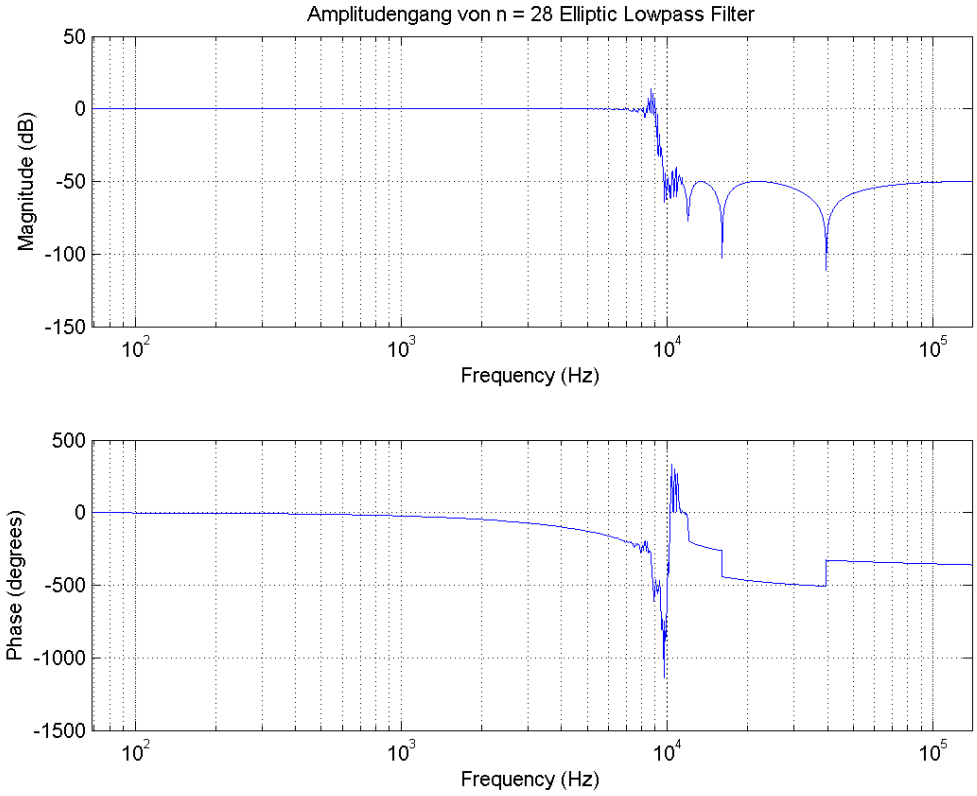


Abbildung 5.12: Elliptischer IIR-Filter

Die folgenden Gleichungen (5.2 und 5.3) beschreiben diese Sektionen.

Für die **Sektion 1** gilt:

$$\boldsymbol{y_1(k) = S_1 \cdot x_1(k) + b_{21} \cdot x_1(k-1) + b_{31} \cdot x_1(k-2)} \\ \boldsymbol{-a_{21} \cdot y_1(k-1) - a_{31} \cdot y_1(k-2)} \tag{5.2}$$

Für die **Sektionen 2-7** gilt:

$$\boldsymbol{y_i(k) = y_j(k) + b_{2i} \cdot y_j(k-1) + b_{3i} \cdot y_j(k-2)} \\ \boldsymbol{-a_{2i} \cdot y_i(k-1) - a_{3i} \cdot y_i(k-2)} \tag{5.3}$$

mit $y_1(k)$ – aktuelles Ergebnis der Sektion 1; $x_1(k)$ – Eingang *Upos* zum Zeitpunkt t=0; $y_i(k)$ – aktuelles Ergebnis der Sektion i (wobei $i=2\ldots7$); $y_j(k)$ – aktuelles Ergebnis der Sektion (i-1); S_1, a_{2i}, a_{3i}, b_{2i}, b_{3i} – Koeffizienten des Elliptischen Filters.

Zur Implementierung des Elliptischen Filters auf dem FPGA wurden die im Abschnitt 4 beschriebenen Analysemethoden verwendet. Dabei wurde die Gesamtstruktur des Filters (entsprechend der Abb. A.2 im Anhang A.1 und Gl. 5.2 und 5.3) mit den speziell für die

Arbeit entwickelten Bibliotheken in LabVIEW realisiert. Das entstandene Datenflussmodell enthält insgesamt 39 Multiplikationen, 14 Additionen und 14 Subtraktionen, die zur Erreichung der notwendigen Rechengenauigkeit mit Fließkomma-Arithmetik doppelter Genauigkeit realisiert werden müssen. Die direkte vollständig parallele Umsetzung auf dem FPGA ist auf Grund der benötigten Ressourcen nicht realisierbar. Eine mögliche Reduzierung auf Kosten der Latenzzeiten besteht in der mehrfachen Verwendung von gleichen Ressourcen des FPGA-Chips (s. Abschnitt 4.4.2). In diesem Fall werden lokale Steuer- und zusätzliche Datenflüsse erzeugt, die jeweils Sequenzen für die Verwendung des gleichen Hardware-Elements an unterschiedlichen Stellen des Berechnungsweges realisieren. Basierend auf der optimierten Datenflussmodellumsetzung wurde eine Struktur des Elliptischen Filters zur Berechnung auf dem FPGA erstellt. Diese enthält 3 Multiplikationen, 1 Subtraktion und 2 Additionen.

Die gemessene Laufzeit dieser Realisierung ist 140 Takte (umgerechnet 3,5 µs), dabei entspricht dieses der gewünschten Anforderung für die maximal mögliche Abtastfrequenz von 280 kHz.

Die beschriebenen Realisierungsmöglichkeiten eines digitalen Vorfilters wurden für unterschiedliche Varianten des Regelungssystems untersucht.

Im Falle der Verwendung der ersten Vorfilter-Variante (FIR-Filter als „Downsampling"-Funktion mit dem Mittelwertbildner) ist die Abtastung mit unterschiedlichen Frequenzen möglich. Zur Parametrisierung dieser Vorfilter-Variante können die Abtastfrequenz und die Anzahl von aufeinander folgenden Werten *N* vom Nutzer (bzw. Entwickler) geändert werden. Zum Vergleich mit der Variante 2 wurden die Abtastfrequenz 280 kHz und *N=28* gewählt, wobei der Regelkreis bei 10 kHz läuft. Außer höherer Flexibilität zur Anpassung der Geschwindigkeit braucht diese Implementierung auf dem FPGA bedeutend weniger Ressourcen (s. Abb. 5.13).

Die Verwendung der zweiten Vorfilter-Variante realisiert einen Elliptischen IIR-Filter, der durch die Parametrisierung seiner Koeffizienten angepasst werden kann. Die Abtastfrequenz ist aber durch die Laufzeit der Realisierung begrenzt und ermöglicht dabei die Abtastung mit der maximalen Frequenz von 280 kHz. Die Geschwindigkeit des Regelkreises wird in diesem Fall sowohl von der Realisierung des Filters als auch der des Regelungsalgorithmus abhängen (s. Abschnitt 5.2.3.3).

Die Abbildung 5.13 zeigt den Ressourcenverbrauch der Implementierungen der beschriebenen Vorfilterungsvarianten auf dem FPGA Virtex-5-LX110. Die FPGA-Taktfrequenz betrug 40 MHz. Die Ergebnisse wurden in Form eines Diagramms und in der Tabelle 5.1 nach der Compilation zusammengefasst.

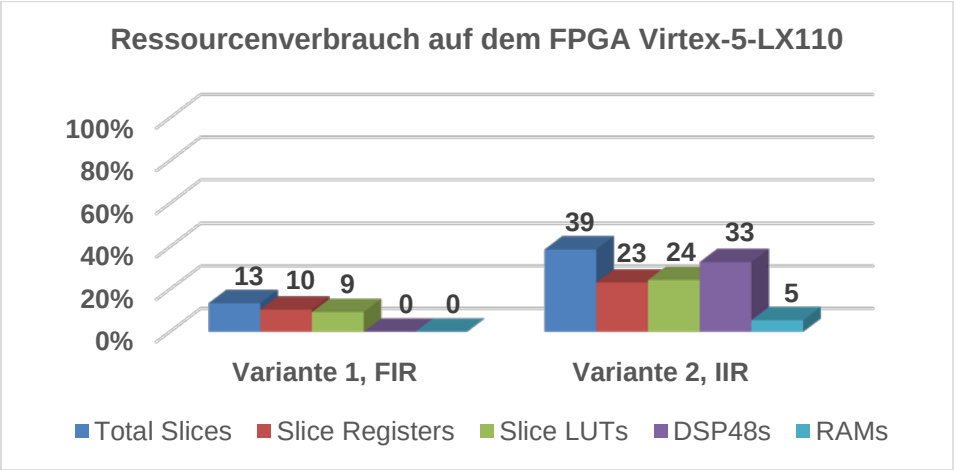


Abbildung 5.13: Ressourcenvergleich der Vorfilter-Implementierungen

	gesamte Ressourcen	Variante 1, FIR	Variante 2, IIR
Total Slices	17280	2322	6689
Slice Registers	69120	7208	16037
Slice LUTs	69120	6420	16865
DSP48s	64	0	21
RAMs	128	0	7

Tabelle 5.1: Ressourcenverbrauch der Vorfilterung auf dem FPGA Virtex-5-LX110

5.2.3.3 Regelungssystem

Dieser Abschnitt stellt die Schritte des Entwurfes des Regelungssystems und der unterschiedlichen Implementierung auf dem FPGA vor. Auch in diesem Teil des Informationsverarbeitungssystems wurden verschiedene Partitionierungsmöglichkeiten untersucht. Entsprechend der Abbildung 5.10 gab es grundsätzlich die Realisierungsvariante durch FPGA-Logik (im minimierten Prototyp) und die Realisierung durch einen speziell dafür entwickelten Softcore-Prozessor mit Softcore-Programm (im erweiterten und finalen Prototyp).

Die modellbasierte Entwicklung unterstützt die Untersuchung und den Entwurf von unterschiedlichen Reglertypen. Beim Entwurf eines Reglers wird das Verhalten des Einbettenden Systems (Mechanik der Wägezelle, Elektronik einschließlich Elektromechanik) von Anfang an in das Modell der Übertragungsfunktion der Strecke des Reglers integriert. Weiterhin gibt es Mechanismen, z.B. spezielle integrierte Funktionen in MATLAB, zur Darstellung der charakteristischen zeitdiskreten Übertragungsfunktion des Reglers abhängig von der gewünschten Abtastfrequenz. Dabei können die Reglerparameter durch Variationen an die entsprechenden messtechnischen Anforderungen (hier: kurze Einschwingzeit, Robustheit) festgelegt werden. Die im Abschnitt 4.3.1.1 ermittelte Struktur des Reglers (s. Gl. 4.3) erlaubt, unterschiedliche Regelungsalgorithmen durch die Parametrisierung von Reglerkoeffizienten zu realisieren.

Basierend auf den Kenntnissen aus dem Vorläufersystem [Ba15][Ng10] und eigenen modellbasierten Untersuchungen konnte eine maximal notwendige Ordnung des Polynoms angepasst werden. Es handelt sich um ein Polynom 10. Ordnung (Gl. 5.4). Das entsprechende LabVIEW Modell befindet sich im Anhang A.5 (Abb. A.12).

$$\mathbf{y(0)} \cdot \mathbf{n(0)} = \big(\mathbf{e(0)} \cdot \mathbf{z(0)} + \mathbf{e(1)} \cdot \mathbf{z(1)} + \cdots + \mathbf{e(9)} \cdot \mathbf{z(9)}\big) - ((\mathbf{y(1)} \cdot \mathbf{n(1)} + \cdots + \mathbf{y(9)} \cdot \mathbf{n(9)}) \tag{5.4}$$

mit *e(0)* - *e(9)* – Regelabweichung zum Zeitpunkt *t; z(0)* – *z(9)* – Koeffizienten des Zählerpolynoms; *y(0)* – *y(9)* – entsprechendes Ergebnis der Regelung (Stellgröße *Ur*); *n(0)* – *n(9)* – Koeffizienten des Nennerpolynoms.

Im Folgenden werden die unterschiedlichen Umsetzungsvarianten auf dem FPGA detaillierter dargestellt.

Im minimierten Prototyp wurde ein Polynomregler 3. Ordnung realisiert. Dadurch konnten PID-Regler mit unterschiedlicher Abtastfrequenz durch die Parametrisierung von Koeffizienten des Zählers (z_0-z_2) und des Nenners (n_0-n_2) umgesetzt werden. Diese Realisierung stellt eine einfache Datenflussstruktur dar und benötigt 5 Multiplikationen, 3 Additionen und eine Subtraktion. Zur Implementierung wurden Realisierungen mit den in LabVIEW zur Verfügung gestellten Fixed-Point-Komponenten und mit den selbst entwickelten Floating-Point-Bibliotheken doppelter Genauigkeit durchgeführt (s. Anhang A.5: Abb. A.12 und A.13).

Der große Vorteil der Fixed-Point-Realisierung ist ein geringerer Ressourcenverbrauch (s. Abbildung 5.14) und eine höhere Geschwindigkeit im Vergleich zur Floating-Point-Realisierung. Bezüglich der Latenz erreichen die beiden Realisierungen die folgende Performance: die Fixed-Point-Realisierung benötigt 5 Takte (0,13 µs bei FPGA-Taktfrequenz 40 MHz) und die Floating-Point-Realisierung 42 Takte (1,05µs bei FPGA-Taktfrequenz 40 MHz).

Das primäre Ergebnis der Regelungsrealisierung im minimierten Prototyp war eine stabile Prinziplösung mit einer möglichen Abtastung von 200 kHz. Dadurch wurde unter anderem der Regelungsalgorithmus aus dem Vorläufersystem umgesetzt, wobei dieser im neuen Prototyp eine deutlich bessere Latenz lieferte. Statt 10 kHz im Vorläufersystem wurde im neuen Prototyp bis zu einer Abtastfrequenz von 300 kHz getestet. Weiterhin konnte während der Untersuchungen festgestellt werden, dass die implementierten Lösungen noch größere Leistungsreserven besitzen und die Regelung durchaus mit einer Abtastfrequenz von 500 kHz betrieben werden kann. Entsprechend [Kr04] führt eine deutliche Abtastfrequenzerhöhung zur Messzeitverkürzung.

Im erweiterten Prototyp und im finalen Prototyp war die Entscheidung bei der Realisierung des Regelungssystems, in der Implementierung einen möglichst flexiblen Algorithmus zu verwenden, der für unterschiedliche Aufgaben angepasst werden konnte. Dabei musste die Möglichkeit bestehen, die Parameter (betragsmäßig sehr kleine und sehr große Werte, z.B. ~10^{40}) darzustellen und die Berechnungen mit hoher Auflösung durchzuführen.

In dieser Arbeit wurden zwei speziell für die messtechnischen Aufgaben entwickelte Realisierungsvarianten mit Fließkomma-Arithmetik dargestellt (s. Abschnitt 4.4.1). Da die Umsetzung des Polynomreglers 10. Ordnung mit den entwickelten Floating-Point-Bibliotheken (durch FPGA-Logik) auf Grund des Ressourcenverbrauchs nicht möglich ist, wurde der Softcore-Prozessor LiSARD dafür verwendet.

Bei der Implementierung des Regelungssystems auf dem FPGA mittels des Softcore-Prozessors LiSARD ergaben sich die Entwurfsschritte und die neue Repartitionierung, die im Abschnitt 4.5 erläutert wurden. Das Softcore-Programm zur Umsetzung des Algorithmus des Regelungssystems wird in dem mit LabVIEW entwickelten Codekonverter als Assemblerprogramm eingegeben (s. Anhang A.5). Dieses Programm wird durch den Codekonverter in Binärcode umgesetzt. Dabei entstehen die zwei Übersetzungs-VIs, die zur Initialisierung von Programm- und Datenspeicher des Softcores auf dem FPGA dienen. Außerdem kann das Debuginterface benutzt werden, um die Daten- und Programmspeicher des PXI-Controllers ohne Neucompilation des FPGA zu ändern und damit verschiedene Varianten der Programme zu debuggen. Die Parametrisierungen werden über die im Softcore definierten Eingänge realisiert.

Die Laufzeit der Implementierung des Regelungssystems ist 1,25 µs. In diesem Fall war der gesamte Ablauf auf dem FPGA mit der Taktfrequenz 40 MHz realisiert, wobei der Rechenkern selbst in eine Single-Cycle-Timed-Loop mit 120 MHz eingebettet wurde. Im erweiterten und im finalen Prototyp wurden die Algorithmen mit der Abtastung von Messsignalen bis zu 270 kHz getestet. Dieses ist durch die Laufzeit der implementierten Funktionen der gesamten Signalverarbeitung begrenzt.

Die Abbildung 5.14 und die Tabelle 5.2 zeigen den Ressourcenverbrauch der beschriebenen Implementierungen des Regelungssystems auf dem FPGA Virtex5-LX110. Dabei sind die unterschiedlich umgesetzten Algorithmen des Polynomreglers dargestellt: Polynom 3. Ordnung (Fixed-Point- und Floating-Point-Realisierungen) und Polynom 10. Ordnung (Softcore-Realisierung).

Es ist wie erwartet zu erkennen, dass die Realisierung mit dem Softcore eine ressourcensparende Lösung ist. In diesem Fall ist zu berücksichtigen, dass ein Grundbedarf an Ressourcen durch den Softcore-Prozessor unabhängig vom Algorithmus entsteht. Man kann daraus schließen, dass auch bei komplexeren Algorithmen die benötigten Gesamtressourcen auf dem FPGA nur geringfügig steigen.

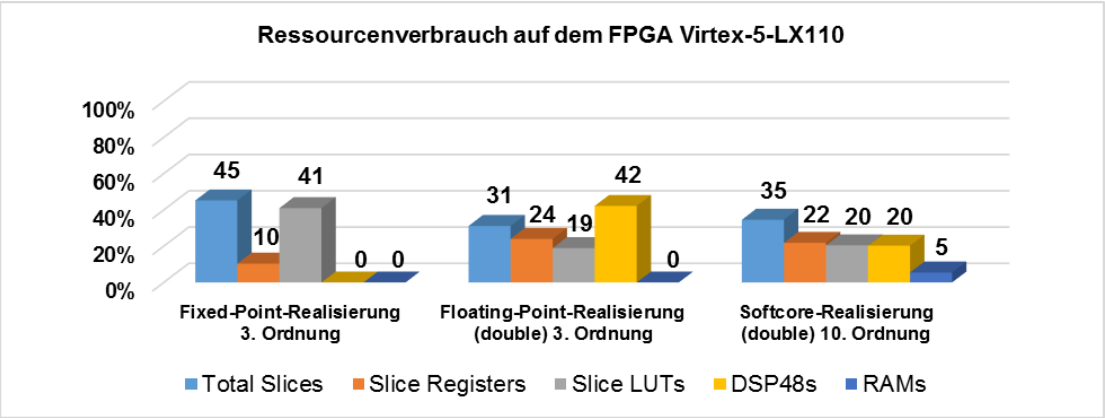


Abbildung 5.14: Ressourcenvergleich von Implementierungen des Regelungssystems

	gesamte Ressourcen	**Fixed-Point Realisierung**	**Floating-Point Realisierung (double)**	**Softcore Realisierung (double)**
Total Slices	17280	7817	5385	5979
Slice Registers	69120	7175	16594	15085
Slice LUTs	69120	28347	13108	14155
DSP48s	64	0	27	13
RAMs	128	0	0	7

Tabelle 5.2: Ressourcenverbrauch des implementierten Regelungssystems

5.2.3.4 Masseberechnung durch Software oder FPGA-basierte Hardware

In diesem Abschnitt wird eine wichtige Funktion der digitalen Informationsverarbeitungskette in der Waage näher dargestellt, die das Messergebnis (die ermittelte Masse) mit der geforderten Messunsicherheit bestimmt. Dabei geht es nicht nur um die direkte Berechnung der Masse aus dem gemessenen Strom (s. Abb. 5.1: über *Usp* vom analogen Eingang ADC1), sondern auch um die Implementierung eines Filters. Es ist hervorzuheben, dass die Qualität des Messergebnisses direkt von der Qualität der Filterung abhängt.

Die vorliegende Arbeit beschäftigt sich nicht mit der Entwicklung eines neuen Filterkonzeptes für die Masseberechnung. Stattdessen wurde das in der Dissertation von Dontsova [Do08] vorgeschlagene und im Vorläufersystem realisierte Konzept im neuen Prototyp umgesetzt. Dabei ging es um den Einsatz von mehreren Summierern mit einem über die gesamte Kaskade operierenden Mittelwertbildner. Dieses Konzept bringt gute Ergebnisse, sowohl im Falle von harmonischen Schwingungen als auch bei verrauschten Signalen. Die Vorteile sind die einfache Realisierbarkeit durch Software, die Möglichkeit zur Erhöhung der Signalauflösung und die Reduzierung von Rundungsfehlern.

Die Abbildung 5.15 zeigt die schematische Datenflussdarstellung der Umsetzung im Projekt für den Prototyp. Zu Gunsten einer übersichtlichen Darstellung wurde hier auf das damit verbundene Zeitscheduling verzichtet. Dieses Scheduling ist notwendig, weil die

Messwerte *Usp* mit jedem Abtastzeitpunkt der Signalverarbeitungsalgorithmen entstehen. Die Masseberechnung benötigt *n* aufeinander folgende Werte, die jeweils mit der Abtastperiode entstehen. Die anschließende Berechnung (Prinzip siehe Abb. 5.15) wird durch eine programmtechnische Realisierung durchgeführt. Da alle Summierer und der abschließende Dividierer gleitend (für jeden Abtastzeitpunkt Berechnungen über die letzten *n* Werte) arbeiten sollen, wurde ein Mechanismus implementiert, der die zeitliche Abhängigkeit der Eingangs- und Zwischenwerte berücksichtigt.

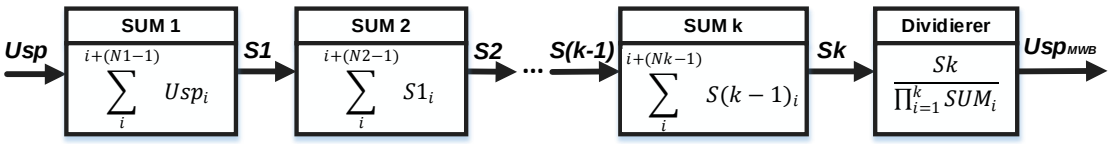


Abbildung 5.15: Filterkonzept des kaskadierten Mittelwertbildners

Der erste gleitende Summierer (SUM 1) addiert *N1* nacheinander folgende Messwerte (*Usp*). Das Ergebnis (entsprechend der Nummer der Kaskade bzw. S_i) wird zu der nächsten Kaskade als Eingang gegeben. Zum Schluss wird das Resultat durch das Produkt aller beteiligten Summiererlängen (Anzahl der für die letzten N_k addierten Werte) dividiert.

In der Dissertation von Dontsova [Do08] wurden unterschiedliche Varianten der digitalen Filterung mit einem oder mehreren kaskadierten Mittelwertbildnern vorgeschlagen und verglichen. Dabei wurden die verschiedenen Längen der Kaskaden analysiert. Auf Basis dieser Untersuchungen wurde in dieser Arbeit ein Filter für die konkrete EMK-Waage entworfen. Der hier verwendete und implementierte digitale Filter besteht aus sieben Kaskaden (beziehungsweise SUM 1-SUM 7). Die Längen der Kaskaden bzw. Summierer sind entsprechend: *N1=20, N2=26, N3=43, N4=54, N5=93, N6=115, N7=512.* Die Länge jeder Kaskade soll aber parametrisierbar bleiben, um die Struktur des Filters an verschiedene Anwendungen anpassen zu können.

Bei der Implementierung der gefilterten und nicht gefilterten Masseberechnung wurde die Partitionierung auf beiden zur Verfügung gestellten Zielplattformen durchgeführt (s. Abb. 5.10). Dabei wird das abgetastete Signal *Usp* vom analogen Eingang ADC1 auf dem FPGA erfasst und zum PXI-Controller transportiert. In diesem Fall erfolgten die Funktionen der Filterung und der Massenberechnung vollständig auf dem PXI-Controller. Diese Funktionen wurden als Software-Programme in LabVIEW realisiert (s. Anhang A.6). Während des Entwicklungsprozesses ist es möglich, die gefilterten und nichtgefilterten Ergebnisse der Masse zu vergleichen, in Form eines Diagramms anzuzeigen und durch die entstehenden Vergleichsergebnisse gezielt Änderungen am Algorithmus zu untersuchen.

5.2.4 Sicherheitsfunktionen für die Metrologische Sicherheit

Die Realisierung von Sicherheitsfunktionen und speziellen Komponenten zur Metrologischen Sicherheit gehört zu den in der vorliegenden Dissertation betrachteten primären Aufgaben. Diese wurden basierend auf den im Abschnitt 4.2.3 beschriebenen Ansätzen in dem entwickelten Informationsverarbeitungssystem umgesetzt. Im Entwicklungsprozess sollen zusätzliche Algorithmen und Schutzmaßnahmen in früheren Entwurfsphasen betrachtet werden. Diese sind als Teil der gesamten Signalverarbeitung zu implementieren. Auf Grund der Forderung der Eichfähigkeit sind vor allem die Funktionen besonders zu sichern, die zur Bildung und Anzeige bzw. Ausgabe des Messergebnisses beitragen. Deswegen müssen die Umsetzungen mit unterschiedlichen Partitionierungen abhängig von den gestellten Anforderungen untersucht werden. Dieser Abschnitt stellt die Arbeiten dar, die im Laufe der Prototypenentwicklung durchgeführt wurden.

Für den erweiterten Prototyp kamen unterschiedliche Funktionen zum allgemeinen Schutz gegen Angriffe und zur Gewährleistung der Metrologischen Sicherheit zum Einsatz. In der Masterarbeit von Herrn Raman [Ra13] wurden die folgenden Komponenten beziehungsweise Funktionen entworfen und über LabVIEW für die verschiedenen Partitionen umgesetzt (s. Abb. 5.16).

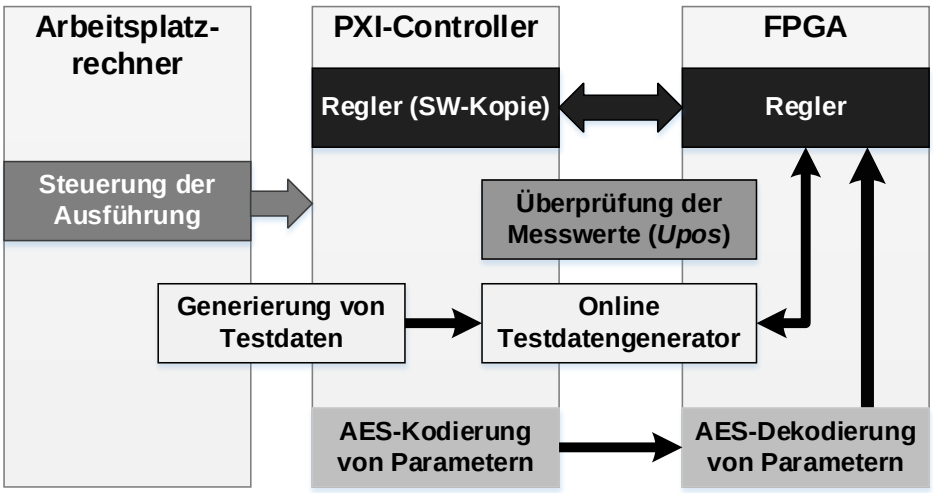


Abbildung 5.16: Realisierte Sicherheitsfunktionen

Das **Redundanzmodul** stellt die Umsetzung des gleichen Regelungsalgorithmus als Software dar, wie er auch auf dem FPGA implementiert wurde. Diese Variante dient zur Kontrolle der richtigen Berechnung des Algorithmus. Dabei werden die Ergebnisse der Berechnung auf dem FPGA mit denen auf dem Controller verglichen. Falls bei der Auswertung Fehler auftreten, kann das einen Ausfall einer der realisierten Lösungen (in Soft- oder Hardware) oder eventuell auch eine unkorrekte Implementierung bedeuten. Zur Gewährleistung der Metrologischen Sicherheit und Eichfähigkeit wird die Ausgabe von Messwerten gestoppt und eine Nachricht darüber dem Nutzer angezeigt.

Die **Überprüfung auf Signifikanz** ist als eine Schutzmaßnahme realisiert, die die abgetasteten Messwerte während der Ausführung mit den erwarteten Werten vergleicht. Dabei wird validiert, ob die Messwerte in definierten Bereichen liegen (hier *Upos* im maximalen Bereich ± 6V). Der PXI-Controller dient zum Vergleich und zur Auswertung. Für den Fall der Feststellung eines Fehlers wird die Messung gestoppt und eine entsprechende Fehlermeldung ausgegeben.

Der **Online-Testdatengenerator** erzeugt geeignete Eingangstestdaten für das Regelungssystem und überprüft die Ergebnisse mit den zugehörigen, vorher berechneten erwarteten Werten. Der Überprüfungsprozess läuft quasiparallel zur eigentlichen Informationsverarbeitung auf dem FPGA. In diesem Fall werden die Aktivierung dieser Funktion und die Erzeugung von Testdaten auf dem Arbeitsrechner ausgeführt. Der auf dem FPGA realisierte Regelungsalgorithmus rechnet zwischen den Abtastzeiten mit den generierten Daten und übermittelt die Ergebnisse zur Auswertung an den PXI-Controller.

Die **AES-Verschlüsselung der übertragenen Informationen** ist eine Maßnahme für die äußere Sicherheit. Dabei werden die Daten gegenüber Veränderung, z.B. durch gewollte Manipulation oder auch Transferfehler, gesichert. In der Applikation wurden beispielhaft die übertragenen Parameter (Koeffizienten des Reglers) auf dem PXI-Controller mittels des AES-Verfahrens kodiert und nach der Übertragung mit dem entsprechenden Schlüssel auf der FPGA-Seite dekodiert und bei der Signalverarbeitung benutzt. Zur Implementierung dieser Funktion in LabVIEW wurden die von National Instruments zur Verfügung gestellten Bibliotheken verwendet. In einer späteren produzierbaren Lösung muss diese Sicherung auf alle extern übertragenen Daten, die zum Messergebnis beitragen, angewendet werden.

Die Tabelle 5.3 stellt die Ergebnisse des Ressourcenverbrauchs der auf dem FPGA umgesetzten Komponenten bzw. Teilkomponenten dar. In diesem Fall ist der Verbrauch für den FPGA Virtex-5-LX85 auf der Karte NI PXI-7853R dargestellt.

	Logik-Ressourcen			**DSP Ressourcen**	**Block RAMs**
	Total Slices	**Slice Registers**	**Slice LUTs**		
insgesamt	12960	51840	51840	48	96
verbraucht	5578	15467	15204	16	37
in %	43,0	29,8	29,3	33,3	38,5

Tabelle 5.3: Ressourcenverbrauch der realisierten Sicherheitsfunktionen

Es ist zu sehen, dass der Gesamtverbrauch über 40% der FPGA-Ressourcen beträgt, wobei die aufwendigste Realisierung die AES-Verschlüsselung ist. Nach [Ra13] ist der maximale Ressourcenverbrauch ohne AES-Algorithmus etwa 12%. Für eine produzierbare Lösung muss insbesondere eine ressourcensparsamere Verschlüsselung genutzt oder entwickelt werden.

Die oben beschriebenen Sicherheitsfunktionen und speziellen Komponenten für die Metrologische Sicherheit wurden im erweiterten Prototyp vollständig und im finalen nur teilweise umgesetzt. Letzterer stellt eine relativ fixierte Informationsverarbeitung dar und nähert sich einer produzierbaren Lösung an. Deswegen sind in diesem die Sicherheitsfunktionen auf den Online-Testdatengenerator und die Überprüfung auf Signifikanz eingeschränkt (s. Abschnitt 5.2.6.1).

5.2.5 Werkzeugunterstütze Validierung und Verifikation in verschiedenen Entwurfsschritten

Die entwurfsbegleitenden Validierung- und Verifikationsaktivitäten sind Bestandteile des Prozesses ZEfIRA. Bei der Entwicklung des Prototyps im Projekt wurde exemplarisch eine Möglichkeit der werkzeugunterstützten Validierung und Verifikation auf unterschiedlichen Abstraktionsebenen untersucht und in der Entwicklungsumgebung LabVIEW umgesetzt.

In der Masterarbeit von Frau Balasubramanyam [Bal14] wurde ein Beispiel (s. Abb. 5.17) für die vorliegende Dissertation realisiert. Dieses basiert auf der Verwendung von mehreren in LabVIEW integrierten Tools. Außer den von National Instruments empfohlenen Tools, wie z.B. NI Unit Test Framework, LabVIEW VI Analyzer und LabVIEW Desktop Execution Trace, die zum Testen in verschiedenen Entwurfsschritten zum Einsatz kommen können, wurde das Tool „LabVIEW Statechart Module“ verwendet. Dabei wurden die einzelnen States durch hierarchisch untergeordnete VIs untersetzt. Das grundliegende Konzept besteht darin, das automatisierte Testen für eine entwickelte Systemkomponente beim modellbasierten Entwurf durchzuführen. Dieses wurde in Form einer Testumgebung realisiert, die eine Schnittstelle zwischen den entwickelten Modellen (hier VIs) und den definierten Testmodulen darstellt. Im Projekt wurden die folgenden Analysen durchgeführt und in Form von Modulen in LabVIEW entwickelt:

- **Statische Code-Analyse**: überprüft die umgesetzten Modelle auf Vollständigkeit und Ausführbarkeit (z.B. dass der Datenfluss keine inkonsistenten Schnittstellen enthält);
- **Unit-Test**: führt eine Black-Box-Prüfung durch, in der die Ergebnisse einer Realisierung mit vordefinierten Werten verglichen werden;
- **Modul zur Analyse der Ausführungszeit**: berechnet die maximale Laufzeit der getesteten Komponenten (hier VI);
- **Analyse und Kontrolle des Speicherbedarfes**: liefert die Information zum notwendigen Speicherverbrauch des Modells auf dem aktuellen und einem eventuellen späteren Zielrechner;

- **Dynamische Code-Analyse**: überprüft die Erfassung von Daten und sucht nach entstehenden Fehlern während der Ausführung;
- **Analyse der Eintrittsinvarianz**: überprüft, ob das getestete VI „reentrant“ oder „non-reentrant“ (s. Abschnitt 4.4.2: Resource-Sharing in LabVIEW) ist und damit einer entsprechenden Forderung danach genügt.

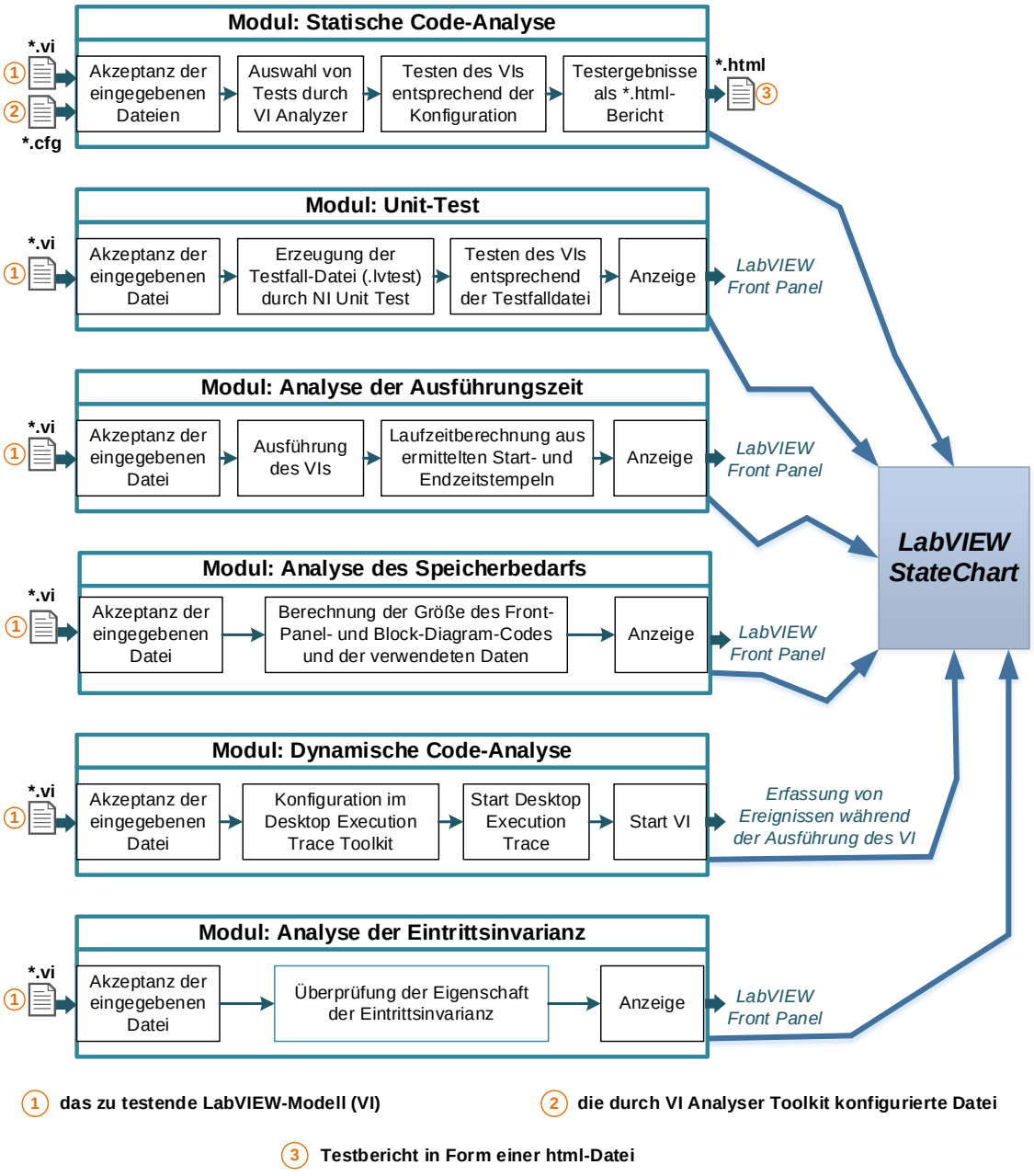


Abbildung 5.17: Realisierung von unterschiedlichen Testmodulen in LabVIEW

Der wichtigste Teil des Konzeptes enthält ein spezielles Statechart-Modell. Dieses mit dem Tool „LabVIEW Statechart Module“ realisierte System (s. Anhang A.7) ermöglicht die Auswahl einer Testaktivität und deren Anwendung auf das zu untersuchende Modell.

In diesem Zusammenhang wurde die Testumgebung auf Basis der beschriebenen Komponenten in LabVIEW realisiert und kann für unterschiedliche Projekte wiederverwendet werden.

5.2.6 Ergebnisse der digitalen Informationsverarbeitung in der Waage

Dieser Abschnitt stellt die Ergebnisse der digitalen Informationsverarbeitung in der betrachteten EMK-Waage entsprechen dem entwickelten finalen Prototyp dar. Dafür werden die FPGA-Implementierung näher erläutert und die realisierten Komponenten der Signalverarbeitung inklusive spezieller Sicherheitsfunktionen und einiger Komponenten der Datenverarbeitung dargestellt.

5.2.6.1 Details zur Implementierung

Die Abbildung 5.18 zeigt die auf der FPGA-Karte NI PXI-7854R implementierte Gesamtstruktur. Diese enthält die Komponenten der Signalerfassung und der digitalen Signalverarbeitung (die Regelung), sowie den Online-Testdatengenerator als eine der hier realisierten Sicherheitsfunktionen. Letzterer wurde mittels FPGA-Logik umgesetzt und überprüft die Funktionalität des Reglers zwischen den Abtastzeiten. Dazu werden die Testdaten vom PXI-Controller über einen FIFO-DMA zum Block „Testdatengenerator“ übertragen. Die Vergleichsergebnisse werden anschließend auf dem Controller ausgewertet. Die Masse wird aus dem erfassten Kompensationsstrom (umgewandelte Spulenspannung *Usp* vom ADC1) auf dem PXI-Controller berechnet und, wenn gewünscht, gefiltert. Zum Datentransfer von Messwerten (Abtastung von drei Kanälen, Zwischenergebnisse der Regelung) mit geringer Latenz zwischen der FPGA-Karte und dem PXI-Controller wurden die weiteren FIFO-DMAs verwendet.

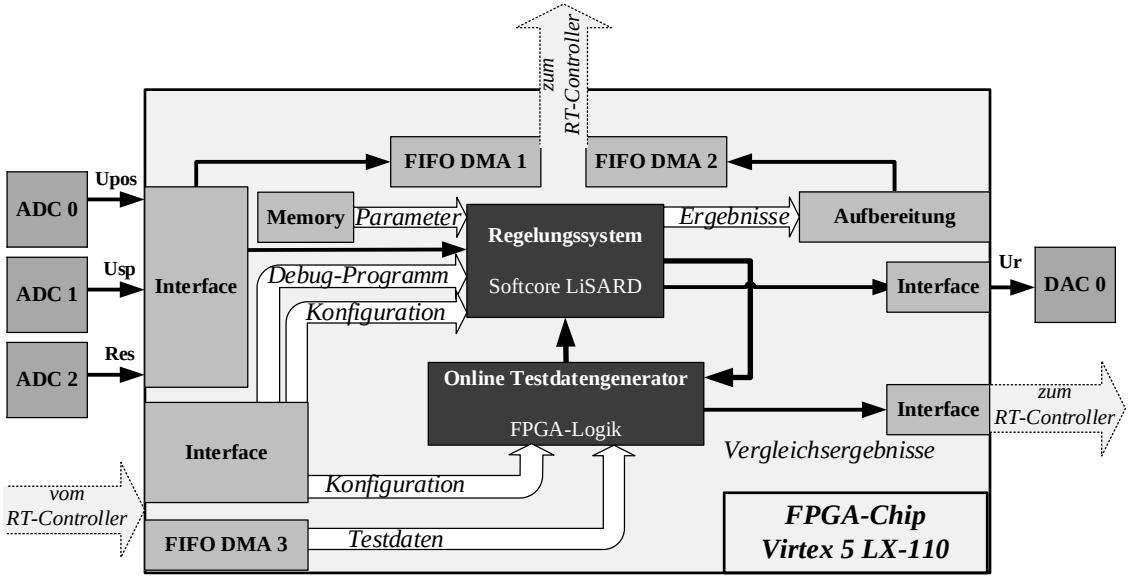


Abbildung 5.18: Schematische Darstellung der FPGA-Implementierung

Der PXI-Controller realisiert hauptsächlich die Parametrisierung der FPGA Programme sowie die Messdatenauswertung. Weiterhin ist es auch möglich, die Festplatte des Controllers zur Abspeicherung der Messdaten zu benutzen. Die graphische Darstellung der Messdaten erfolgt offline nach der Abspeicherung mit LabVIEW oder MATLAB. Außerdem enthält das Hauptprogramm auf dem PXI-Controller neben der Funktion der Masseberechnung zusätzliche Überwachungsfunktionen. Dies bedeutet, dass beim Messen zusätzliche Informationen über den Verlauf beziehungsweise die Ausführung auf dem FPGA ausgewertet werden (z.B. Überwachung der Kommunikation zwischen den Teilkomponenten, des dynamischen Speicherplatzes und des Überlaufens des FIFO-Puffers im DMA-Baustein). Als zusätzliche Maßnahme wurde die Überwachung der Positionsspannung *Upos* implementiert.

Der Hauptteil des realisierten Signalverarbeitungssystems ist das Regelungssystem. Die programmtechnische Umsetzung erfolgte unter Zuhilfenahme des Softcore-Prozessors LiSARD mit den entsprechenden Softcore-Programmen. Der Befehlssatz des Rechenkernes wurde im Sinne eines ASIP (A*pplication Specific Instruction Set Processor*) für die vorliegende Applikation so angepasst, dass nur die in dieser Implementierung notwendigen Operatoren verwendet werden (s. Tabelle A.4 im Anhang A.8). Dieses Vorgehen erzielte eine Optimierung des Ressourcenverbrauchs und stellt damit eine angepasste Lösung dar. Zur Unterstützung der Flexibilität der Anwendung im finalen Prototyp wurden unterschiedliche Varianten der Regelungsalgorithmen umgesetzt.

Zu einem wurde der Regelungsalgorithmus als Polynom-Regler 10 .Ordnung umgesetzt (s. Abschnitt 5.2.3.3). Entsprechend der Konfiguration vom Nutzer (Auswahl des Algorithmus) wird aus dem Speicher (s. Memory-Block in der Abb. 5.18) der zugehörige Parameter- bzw. Koeffizientensatz gelesen. Darauf basierend konnten im Projekt die PID-Regler aus dem Vorläufersystem und ein neu entworfener, sowie Kompensationsregler und IMC-Regler mit unterschiedlichen Abtastfrequenzen analysiert und implementiert werden.

Zu anderem wird in der dargestellten Implementierung das Debuginterface des Softcore-Prozessors LiSARD benutzt, um eine weitere Variante des Regelungsalgorithmus, einen Zustandsregler, umzusetzen. In diesem Fall werden die Daten- und Programmspeicher des Softcores vom PXI-Controller ohne Neucompilation des FPGA geändert und durch die Konfiguration die entsprechenden Parameter aus dem Speicher geladen. Durch die Debugging-Funktion wurde dieser Zustandsregler in der Applikation implementiert. Er wurde in Kooperation mit dem FG Systemanalyse der TU Ilmenau modellbasiert untersucht und entworfen mit dem Ziel, einen schnelleren Stellgrößenverlauf zu erhalten.

Der Zustandsregler beinhaltet einen PI-Algorithmus und eine Störgrößenkompensation. Die für die Regelung benötigten Zustände werden mittels eines Beobachters geschätzt. Zur Umwandlung der elektrischen Signale in die mechanischen Größen wurden auch, wie

im oben beschriebenen Regelungssystem, Faktoren basierend auf den Kennlinien von Komponenten des Einbettenden Systems verwendet. Eine Übersicht der Gesamtstruktur gibt die Abbildung 5.19.

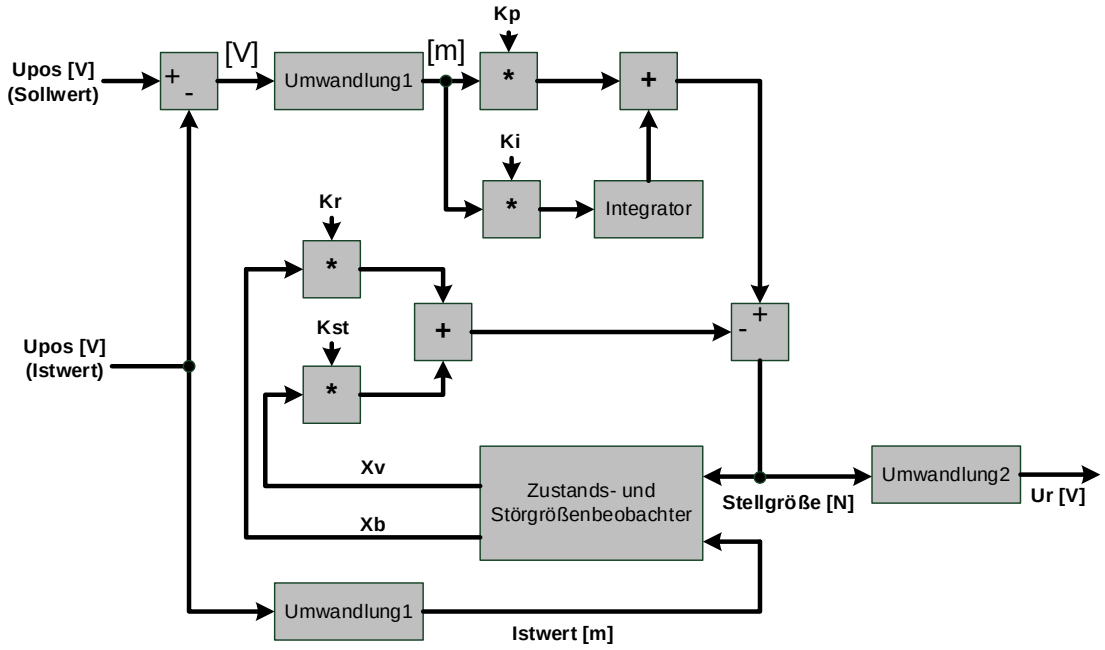


Abbildung 5.19: Struktur des implementierten Zustandsreglers

Die Regeldifferenz wird mit den proportional und integral wirkenden Reglerverstärkungen (K_p und K_i) multipliziert. Eine weitere Komponente, die zur Bildung des Stellsignals beiträgt, ist die Zustandsrückführung über K_r. Ebenso wird eine statische Störgrößenkompensation in Form von K_{st} verwendet. Der Zustand x_v beschreibt dabei die Kraft auf die Waagschale als geschätzte Störgröße.

Die Abbildung 5.20 stellt das realisierte LabVIEW-Programm zur Abbildung 5.18 dar und die Tabelle 5.4 zeigt den Ressourcenverbrauch der beschriebenen Implementierung auf dem FPGA Virtex5-LX110. Der Datenfluss des Regelungsalgorithmus ist in diesem Programm strukturell nicht direkt erkennbar, da dessen Realisierung der Softcore-Prozessor übernimmt.

	Logik-Ressourcen			**DSP-Ressourcen**	**Block RAMs**
	Total Slices	**Slice Registers**	**Slice LUTs**		
insgesamt	17280	69120	69120	64	128
verbraucht	12390	41541	27372	13	25
in %	71,7	60,1	39,6	20,3	19,5

Tabelle 5.4: Ressourcenverbrauch des Informationsverarbeitungssystems

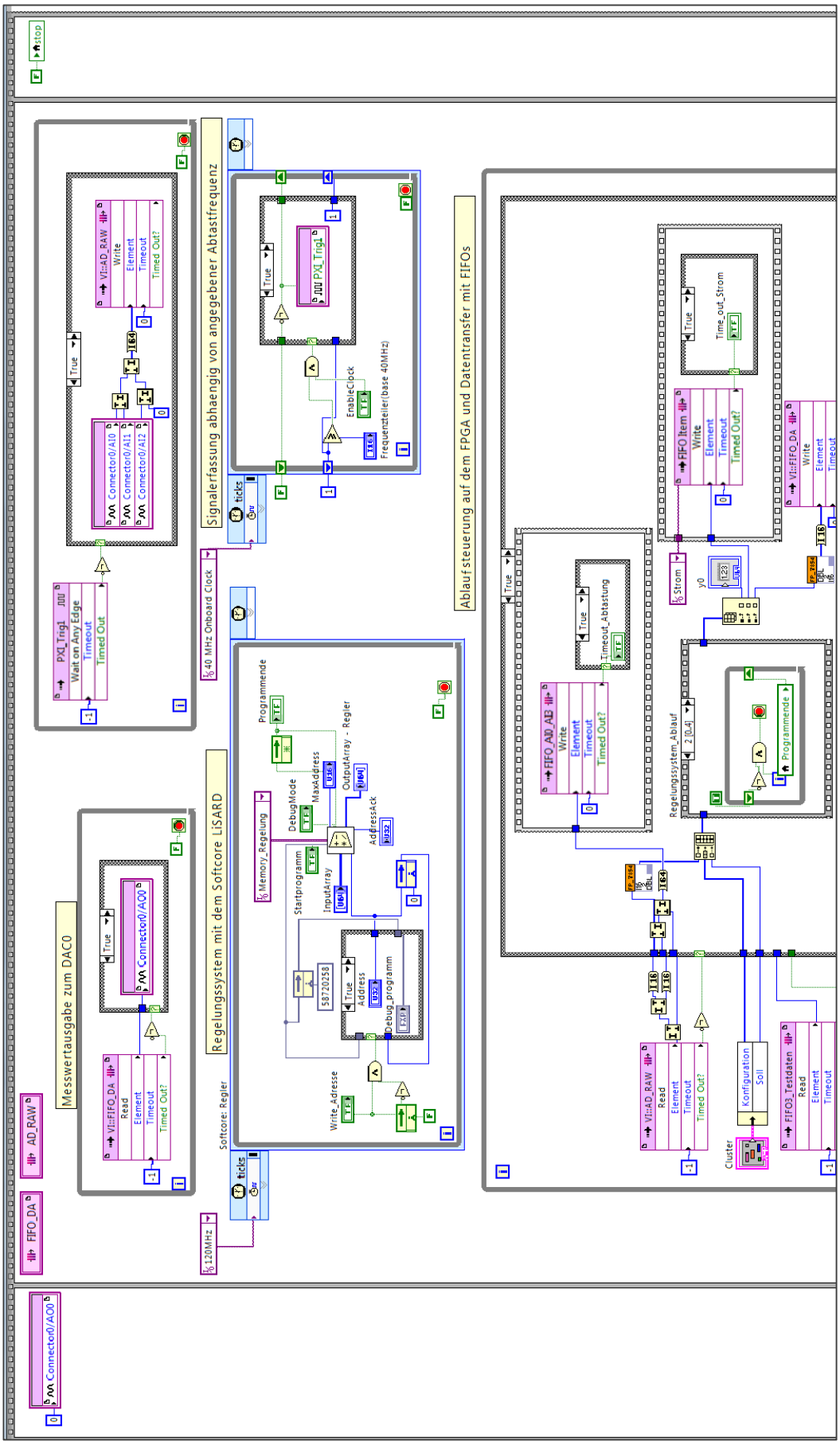


Abbildung 5.20: LabVIEW-Programm der Implementierung auf dem FPGA

Zusammenfassend kann man werten, dass der finale Prototyp durch die Orientierung auf die FPGA-Plattform eine sehr flexible und leistungsfähige Lösung darstellt. Diese ermöglicht die Untersuchung verschieden komplexer Algorithmen mit unterschiedlichen Leistungsparametern. Dabei spielen auch die geringen Latenzen eine Rolle, wobei die Forderungen für die produzierbare Lösung trotzdem unter den hier genutzten liegen können. Gleiches gilt z.B. auch für die verwendeten Genauigkeiten und den Softcore-Einsatz. Weiterhin ist zu berücksichtigen, dass für die Analyse des Einbettenden Systems und des Verhaltens der verschiedenen Algorithmen eine Reihe von Zusatzfunktionen parallel zu den Anwendungsfunktionen arbeiten muss. Diese stellen zusätzliche Anforderungen an die Informationsverarbeitung bezüglich Ressourcen, Zeitbedarf und Systemarchitektur.

5.2.6.2 Messergebnisse

Zur Beurteilung des implementierten Signalverarbeitungssystems wurden Tests mit Hilfe des Lorenzkraftlastwechslers [Ba15] durchgeführt. Dazu wurde mittels eines Funktionsgenerators (hier HP3245A) der Spulenstrom des Lastwechslers rechteckförmig mit Frequenzen von 1 Hz bis 60 Hz moduliert. Die Anregung in Form eines Rechtecksignals symmetrisch zur Nullachse, ist mit einer negativen Kraft (verringertes Gewicht) und das Drücken des Lastwechslers mit einer positiven Kraft (erhöhtes Gewicht) verbunden. Die Ergebnisse der Messung sind ausschnittsweise im Anhang A.9 angegeben. Als Beispiel ist hierzu der Zeitverlauf der Messwerte für den Positionssensor (Positionsspannung *Upos*) in der Abbildung 5.21 und für die Spule (Spulenspannung *Usp*) während einer Lastwechselfolge von 1 Hz in der Abbildung 5.22 dargestellt.

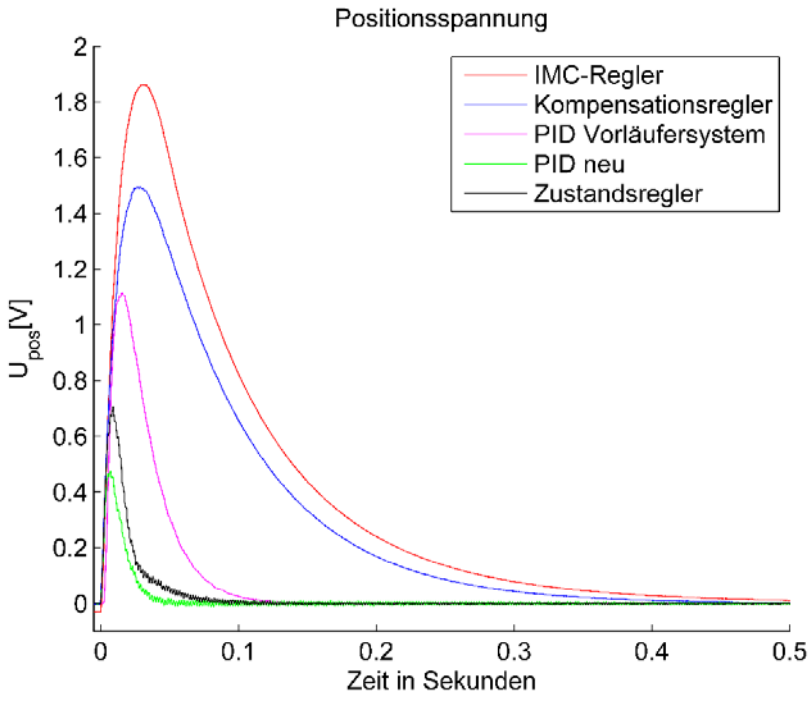


Abbildung 5.21: Positionsspannung während einer Lastwechselfolge mit 1 Hz

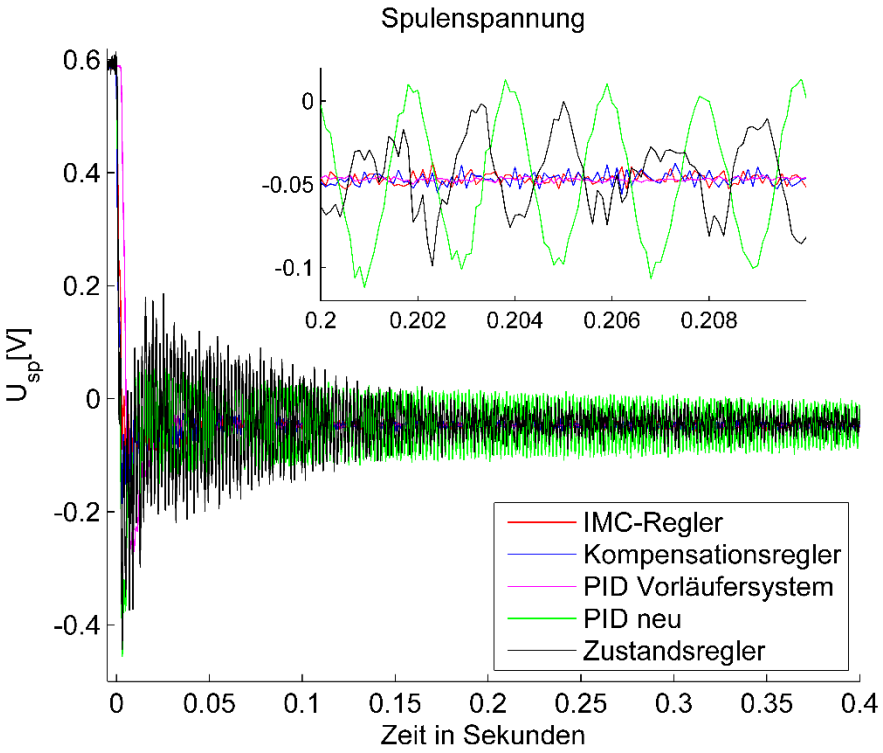


Abbildung 5.22: Spulenspannung während einer Lastwechselfolge mit 1 Hz

In der Abbildung 5.22 ist ein vergrößerter Ausschnitt enthalten, wobei sowohl die Spannungs- als auch die Zeitachsen gedehnt und gegenüber dem Nullpunkt verschoben sind. Dadurch ist der Vergleich der einzelnen Reglervarianten klarer ersichtlich.

Die untersuchten Regelungsalgorithmen sind mit unterschiedlichen Farben abgebildet. Die IMC-, Kompensations- und PID-Regler wurden auf Basis des Polynomreglers (10.Ordnung) umgesetzt. Der realisierte Zustandsregler entspricht der Struktur in der Abbildung 5.19. Die Abtastfrequenz betrug in diesem Fall 10 kHz. Zum Vergleich der Leistungsfähigkeit der Regler wurden weitere Tests bis 300 kHz erfolgreich durchgeführt.

Die PID-Regler und der Zustandsregler zeigen eine bessere Performance bezüglich der Einschwingzeit, wobei der Zustandsregler ein aggressives Verhalten hat. Bei der Bewertung der Regler ist es wichtig zu sagen, dass ein schnelles Ausregeln der Hebelposition nicht zwangsläufig eine bessere Massenbestimmung zur Folge hat. Dieses Phänomen kann z.B. anhand des IMC-Reglers beobachtet werden, denn dieser zeigt gegenüber anderen Reglern ein gutes Verhalten bezüglich des Stellgrößenverlaufes (*Ur*), aus dem die Masse abgeleitet wird (s. Abb. 5.23). Andererseits benötigt er mehr Zeit, um den Hebel in die Nulllage zu bringen (s. Abb. 5.21).

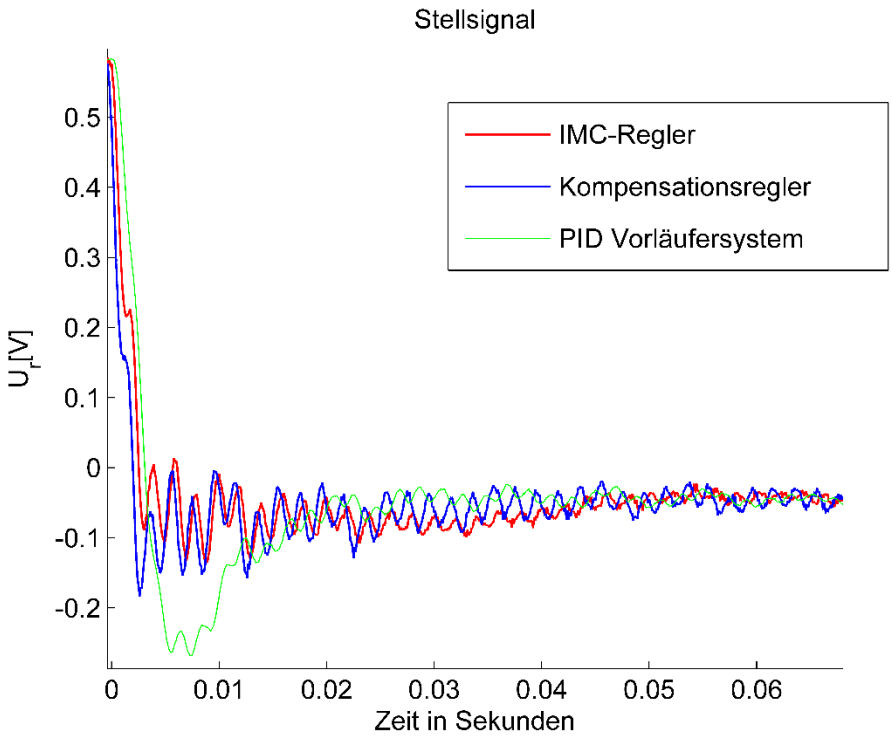


Abbildung 5.23: Vergleich des Stellsignals (ohne PID_{neu} und Zustandsregler)

Die Abbildung 5.24 stellt die Ergebnisse der ermittelten Masse bei einer Lastwechselfolge (1 Hz) dar.

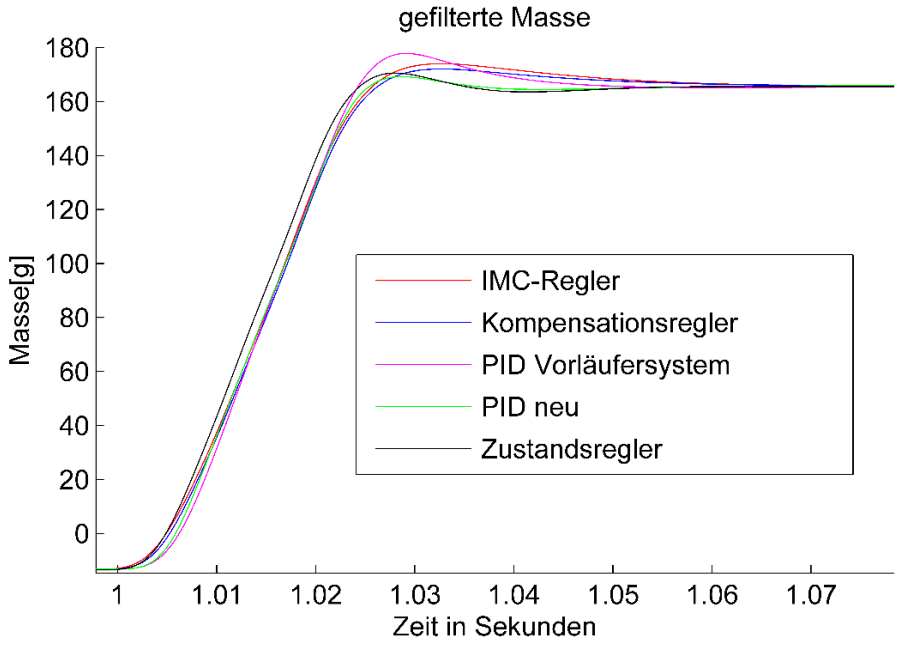


Abbildung 5.24: Ermittelte Masseergebnisse (Filterstufe 5)

Bei der Ermittlung der Masse wurden die Messwerte der Spulenspannung *Usp (*der zum Spulenstrom proportionalen Messspannung) benutzt. Dabei ist die Anwendung eines Filters zur Steigerung der Masseauflösung notwendig, was man an dem Einschwingverhalten der Signale in den Abbildungen 5.22 und 5.23 erkennt. Dafür wurden verschiedene Stufen des kaskadierten Mittelwertbildners (s. Abschnitt 5.2.3.4) ausprobiert, um einen Kompromiss zwischen der Einschwingzeit und dem Signal-Rausch-Verhältnis zu finden. Als Ergebnis wurde die Filterstufe 5 gewählt.

Für verschiedene Regler und Lastwechselfrequenzen wurde jeweils mit einer Lastwechselfolge die Standardabweichung der Massedifferenz bestimmt (Tabelle 5.5). Als wahrer Wert der Massedifferenz wurde dabei der Mittelwert der Lastwechselfolge mit 1 Hz Lastwechselfrequenz angenommen.

Regler	**Belastungszeit**					
	500 ms	**100 ms**	**50 ms**	**25 ms**	**12,5 ms**	**8,3 ms**
IMC-Regler	± 6,14 mg	-0,59 g ± 15,68 mg	6,417 g ± 20,82 mg	-37,78 g ± 282,14 mg	-214,94 g ± 327,65 mg	-177,79 g ± 123,64 mg
Kompensationsregler	± 4,89 mg	-0,42 g ± 19,77 mg	4,34 g ± 19,59 mg	-31,87 g ± 264,96 mg	-214,78 g ± 277,11 mg	-175,08 g ± 135,14 mg
PID, Vorläufersystem	± 12,39 mg	-0,49 g ± 14,04 mg	0,03 g ± 33,20 mg	-10,54 g ± 365,26 mg	-227,74 g ± 520,57 mg	-183,36 g ± 209,95 mg
PID, neu	± 8,19 mg	-0,14 g ± 21,34 mg	-2,36 g ± 40,18 mg	-7,19 g ± 243,94 mg	-217,72 g ± 258,25 mg	-167,89 g ± 294,67 mg
Zustandsregler	± 3,54 mg	-0,07 g ± 15,14 mg	-4,15 g ± 16,38 mg	-0,97 g ± 191,42 mg	-227,99 g ± 609,55 mg	-177,78 g ± 161,74 mg

Tabelle 5.5: Abweichungen der ermittelten Massedifferenzen, Filterstufe 5

Die prototypischen Untersuchungen zur Informationsverarbeitung (Eingebettetes System) besaßen als nicht veränderbare Restriktionen ein vorgegebenes und nicht veränderbares Einbettendes System (Wägezelle mit deren Elektronik). Naturgemäß können in diesem Kontext durch die Informationsverarbeitung allein nur begrenzte Verbesserungen des Gesamtsystemverhaltens erreicht werden, die die Möglichkeiten des übrigen Systems weitgehend nutzen. Diese Grenzen können, wie in ZEfIRA schon zugrunde gelegt, nur durch die Betrachtung und die Optimierung des Gesamtsystems, sowohl modellbasiert als auch in den realen Aufbauten, erfolgreich sein.

Da während der Untersuchungen eine Begrenzung der erreichbaren Reproduzierbarkeit von etwa 10 mg festgestellt wurde, erscheint es sinnvoll, die weiteren Verbesserungen beziehungsweise Optimierungen des Einbettenden Systems im Zusammenhang mit der Informationsverarbeitung durchzuführen. Dazu schafft der erweiterte Prototyp eine sehr gut geeignete Untersuchungsplattform. In der Algorithmik könnten dann eventuell zur Verbesserung der Regelung ein Kalman-Filter und eine dynamische Störkompensationen betrachtet werden. Des Weiteren sind neue Regelungskonzepte, beispielsweise adaptive

Regler, eine Möglichkeit, die Messdynamik und -genauigkeit zu verbessern. Adaptive Regler sind insofern sinnvoll, als sie sich während des Betriebs auf aktuelle Umgebungsbedingungen einstellen. Für den Prozess ZEfIRA und den entworfenen evolutionären Prototyp bedeutet dieses eine Anpassung beziehungsweise Änderung von einigen entwickelten Modellen und eventuell von Komponenten des Informationsverarbeitungssystems.

5.3 Einschätzung der Effektivität von ZEfIRA

Der entwickelte und beschriebene Prozess ZEfIRA als eine spezielle Vorgehensweise bei der Entwicklung von komplexen Informationsverarbeitungssystemen in der Messtechnik wird in diesem Abschnitt bezüglich seiner Effektivität eingeschätzt. Die Verwendung von ZEfIRA wurde durch die Notwendigkeit einer systematischen und formalisierten Entwicklung inklusive Testaktivitäten motiviert. Der Entwurfsprozess des prototypischen Systems für die Informationsverarbeitungsaufgaben (W-Prozess: „Prototyp“) stand im Vordergrund. In diesem Fall wurde die Prototypenentwicklung vorwiegend zur Untersuchung von unterschiedlichen Lösungsvarianten bezüglich der weitgehend wirtschaftlich und technisch optimalen Realisierung der funktionalen Eigenschaften ausgelegt und benutzt. Ein für diese Zwecke entwickelter Prototyp ermöglicht einen frühzeitigen Nachweis von spezifischen Produkteigenschaften und eine Spezifizierung von unklaren Anforderungen, und er dient zur Demonstration und Validierung in einem messtechnischen Gesamtsystem.

Im beschriebenen Anwendungsprojekt zur EMK-Waage konnten aus mehreren Gründen nicht alle Schritte von ZEfIRA vollständig umgesetzt werden. Zum einen wurde der Entwurf des Einbettenden und des Eingebetteten Systems nicht gemeinsam durchgeführt, da die Arbeit von unterschiedlichen Entwicklungsgruppen zeitversetzt stattfand. In diesem Zusammenhang konnten praktisch keine Wechselwirkungen untereinander bei der Entwicklung der beiden Systeme im Prozess untersucht werden. Zu anderem wurde das Vorgehen des dritten W-Modells „Produktentwicklung“ nur ansatzweise untersucht. Dabei wurden die Prinzipien zur teiloptimierten beziehungsweise optimalen Übertragung von Lösungen auf Basis von entwickelten Prototypen zu kostengünstigeren Produktlösungen mit FPGAs oder ASICs behandelt (s. [Kir13]). Momentan ist die untersuchte FPGA-Lösung von Plattformen des Herstellers National Instruments (PXI-Systeme mit einer zusätzlichen multifunktionalen FPGA-Karte) abhängig. Außerdem wurden die Prototypen in der Entwicklungsumgebung LabVIEW umgesetzt. Zur Übernahme von entworfenen Komponenten aus dem Prototyp bei der Entwicklung einer Produktlösung sollten diese als HDL-Code zur Verfügung stehen, um einen effektiven Zugang zu preisgünstigeren Lösungen zu ermöglichen. In diesem Kontext wurde in der Dissertation die Entwicklung des Softcore-Prozessors ViSARD motiviert. Dieser ist bei gleichem Funktionsumfang

eine kostengünstigere und ressourcensparende Alternative zu dem in den Prototypen umgesetzten LiSARD-Konzept [Kir14].

Die Untersuchung der Effektivität von ZEfIRA wird im Folgenden aspektweise durchgeführt:

- Die parallele Durchführung der Analyse des Vorläufersystems und der modellbasierten Entwicklung eines neuen Prototyps wird unterstützt. Durch die Verwendung einer FPGA-Plattform können beide Funktionen flexibel implementiert werden (z.B. wie im erweiterten Prototyp die Funktionen der Systemidentifikation und der Signalverarbeitung). Zusätzlich ist es möglich, die Algorithmen des Vorläufersystems zum Vergleich, jetzt auf dem FPGA, erneut zu implementieren.
- Die Untersuchung von unterschiedlichen Lösungsvarianten (mit verschiedenen Partitionierungen) im Prototyp mit Verwendung eines FPGA und einer programmierbaren Einheit (z.B. Real-Time Controller) ermöglicht eine effektive Analyse. Darauf basierend können Entscheidungen für das produzierbare Erzeugnis abgeleitet werden.
- Der Zugang zu einer hohen Wiederverwendbarkeit von entwickelten Komponenten ermöglicht deren Anwendung in unterschiedlichen Applikationen der Messtechnik. Das ist auch dadurch begründet, dass viele Probleme in der Messtechnik, wie z.B. Signalverarbeitung aber auch Metrologische Sicherheit und Eichfähigkeit, oft ähnlich gelöst werden müssen.
- Entsprechend dem Entwicklungsprozess sind die frühen Testphasen auf hierarchisch aufeinander aufbauenden Modellen und die Ableitung der Testfälle des implementierten Systems auf Basis dieser modellbasierten Tests möglich. Die werkzeugunterstützte Durchführung systematisiert und automatisiert die Ausführung der notwendigen Tests.
- Die Berücksichtigung von gesetzlichen Anforderungen (Eichfähigkeit, Richtlinien, Standards) und der in der Arbeit definierten Metrologischen Sicherheit in allen Entwurfs- und Testetappen der Prototypenentwicklung führt zu einem zertifizierten Produkt, unter anderem durch den zertifizierbaren Prozess.
- Durch das systematische Vorgehen und die Dokumentierung wird eine Effektivitätssteigerung beim Entwurf erzielt und dadurch die Entwicklungszeit verkürzt.

Zum Vergleich von ZEfIRA mit verbreiteten und oft eingesetzten Entwicklungsprozessen (z.B. nach dem V-Vorgehen) wird der abgeschätzte zeitliche Verlauf in der Abbildung 5.25 gezeigt.

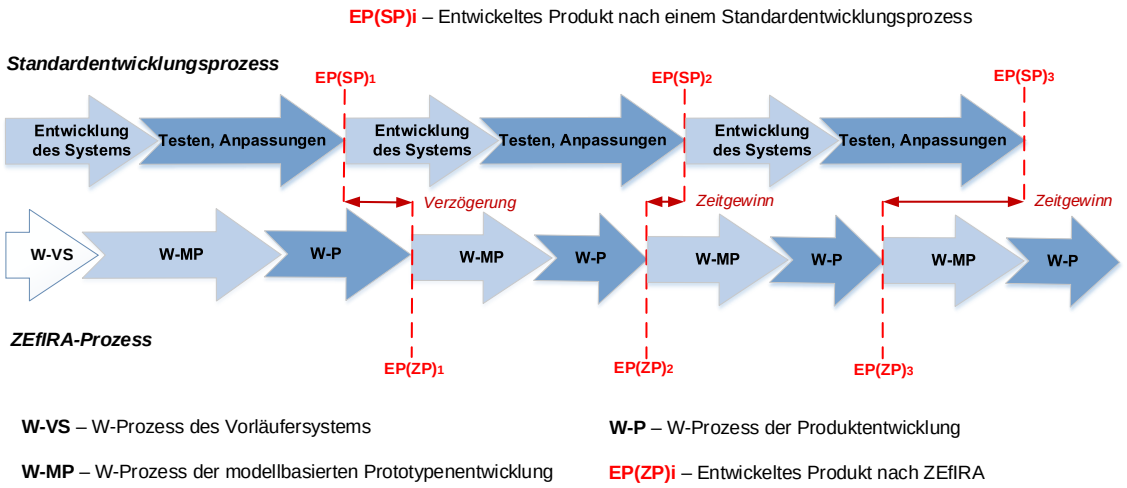


Abbildung 5.25: Vergleich des ZEfIRA- und eines Standardentwicklungsprozesses

Der erste in ZEfIRA laufende Prozess ist die Analyse des Vorläufersystems (W-VS) bezogen auf die Nutzung für die Prototypenentwicklung des aktuellen Erzeugnisses. Dabei kommt es vor allem darauf an, dass die Modelle und Entwurfsaktivitäten als Vorbild für die Prototypenentwicklung (W-MP) dienen können. Je weiter diese in Charakter und Dokumentation divergieren, umso höher kann der benötigte Zeitaufwand sein. Das ist umso mehr der Fall, wenn das Vorläufersystem mit einem Entwicklungsprozess entworfen und dokumentiert wurde, der eine wenig systematische Vorgehensweise, zumeist ohne durchgängige Nutzung von Modellen, beinhaltete. In dem in den Abschnitten 5.1 und 5.2 beschriebenen Anwendungsprojekt waren für die Analyse des Vorläufersystems etwa zwei Drittel des Entwurfsaufwandes eines Standardprozesses notwendig.

Nach der Vorläufersystemanalyse beginnt die Prototypenentwicklung (W-MP). Die Entwicklung eines weitgehend neuen Prototyps (z.B. bei der Nutzung anderer Zielplattformen und Entwicklungsumgebungen) verursacht aufgrund der Ersteinrichtung der Prototyp-Umgebung einen weiteren Aufwand. Dabei kann aber trotzdem eine Verkürzung der aktuellen Entwicklungszeit inklusive Testaktivitäten entstehen, die aus der Verfügbarkeit passender Lösungen und Dokumentationen des Vorläufersystems nach deren Be- und Erarbeitung resultiert. Der Zyklus der Produktentwicklung entsprechend ZEfIRA basiert anschließend auf den Ergebnissen der Prototypenentwicklung. Die resultierende Gesamtzeit der Entwicklung eines neuen Erzeugnisses setzt sich aus der Summe der Zeit für die Analyse, Aufbereitung und Dokumentation des Vorläufersystems und der Entwicklungszeit von Prototyp und Produkt zusammen. Nach einer Abschätzung auf Basis des zugrundeliegenden Anwendungsprojektes wird der Entwicklungsabschluss des ersten Produkts unter der Nutzung von 3W-Vorgehensweisen ($EP(ZP)_1$) zu einem Zeitpunkt liegen, in dem mit einem Standardentwicklungsprozess das zweite entwickelte Produkt ($EP(SP)_2$) schon in Arbeit ist. Diese Verzögerung ist eventuell dadurch

vermeidbar, dass die Analyse des Vorläufersystems und die Aktivitäten der Prototypentwicklung teilweise parallel erfolgen. In diesem Zeitraum wird dann eine höhere Entwicklungskapazität benötigt. Der entscheidende Vorteil des ZEfIRA-Prozesses beginnt mit der nächsten Generation eines Produkts und setzt sich danach weiter fort. Da viele im Prototyp entworfene Komponenten, Methoden und Verfahren wiederverwendet werden können, verkürzen sich die Zeiten der neuen Prototypenentwicklung und der nachfolgenden Produktentwicklung. Im diesem Zusammenhang wird es möglich, $EP(ZP)_2$ vor $EP(SP)_2$ zu erreichen. Bei der Fortsetzung dieses Vorteils ist es denkbar, die folgenden Entwicklungsabschlüsse $EP(ZP)_i$ mit jeweils weiter verkürzten Entwicklungszeiten gegenüber dem Standardentwicklungsprozess zu erreichen.

6 Ansätze zum Produktlinienentwurf in ZEfIRA

Ein ansatzweise bearbeiteter Aspekt der vorliegenden Dissertation beschäftigt sich mit der Erweiterung des Konzeptes ZEfIRA für den Produktlinienentwurf. Der Fokus liegt hierbei in der Realisierung verschiedener Produkte, die ein gemeinsames Basissystem und anwendungsabhängige unterschiedliche Erweiterungen besitzen. Die Komplexität des algorithmischen Teils, die realisierbaren Echtzeiteigenschaften und der Ressourcenverbrauch sind dabei abhängig von der konkreten Ausprägung des Produkts in der Produktlinie. Damit ist eine Produktlinientechnologie ein effektiver Ansatz für eine kostengünstige und kurze Entwicklung und Produktion von einzelnen Erzeugnissen in den zu betrachtenden Anwendungsdomänen.

Das Prinzip einer Produktlinienentwicklung kann schon bei der Entwicklung eines Prototyps angewendet werden. Das ermöglicht, umfangreiche Varianten und Kombinationen von Teilfunktionen mit unterschiedlichen Parametrisierungen zu untersuchen. Dabei entstehen gleichermaßen wie bei der klassischen Produktlinie Abhängigkeiten zwischen den kombinierbaren Teilen und deren Parametern. Beispielsweise können sich bestimmte Teilfunktionen beziehungsweise Features bedingen oder auch ausschließen.

Die in der Arbeit entworfenen wiederverwendbaren modularen Einzelkomponenten des Informationsverarbeitungssystems können für unterschiedliche Anwendungen (bzw. EMK-Waagen) abhängig von den gestellten Anforderungen zusammengesetzt werden. Allen abgeleiteten Anwendungslösungen ist gemeinsam, dass durch die Informationsverarbeitung das EMK-Prinzip von Seiten der Signalverarbeitung einschließlich der Regelung unterstützt wird. Dabei treten Varianten mit unterschiedlichen Leistungskennwerten und Parametern, aber auch mit zusätzlichen Funktionen, abhängig vom Einsatzzweck, auf. Dieser kann sehr unterschiedlich sein, z.B. eine manuell bedienbare Laborwaage oder völlig andersartig, eine Dosierwaage in einer automatisierten Fertigungslinie. Im entwickelten Prototyp sollten dabei alle möglichen Funktionen und deren mögliche Kombinationen und Parameterräume untersucht werden. Dadurch können Produkte mit verschiedenen Merkmalen abgeleitet und für eine Produktlinie benutzt werden. Beim Übergang vom Prototyp zur Produktlinie der produzierbaren Lösungen muss dabei realisiert werden, dass das gemeinsame Basissystem weitgehend unverändert für alle abgeleiteten Anwendungslösungen verwendbar ist. Eine wichtige Rolle spielt dabei eine große Variabilität, die auch in diesem Basissystem verfügbar sein muss. Diese kann beispielweise innerhalb der Architektur durch parametrisierbare Komponenten erreicht werden. Demgegenüber spiegelt sich die Variabilität der verschiedenen zusätzlichen Funktionen vorwiegend in austauschbaren Komponenten wieder.

Die Abbildung 6.1 zeigt ein Feature Diagramm, das auf der Variabilität der entwickelten Prototypen basiert, wobei zur besseren Übersicht nicht alle Komponenten mit ihren Details dargestellt wurden. Dieses Diagramm kann genutzt werden, um mögliche

Gestaltungen eines oder mehrerer Basissysteme zu ermitteln und zu untersuchen und darauf aufbauend die Ausprägung für das produzierbare Erzeugnis abzuleiten. Die benötigten Features einer abgeleiteten konkreten Lösung resultieren aus den Anforderungen an das Basissystem, in dem diese eingesetzt werden soll. Wenn es sich beispielsweise um die Applikationsklasse „Waagen mit geringen Anforderungen an die Messunsicherheit“ handelt, kann daraus beispielsweise ein Feature „Festkomma-Arithmetik statt Gleitkomma-Arithmetik“ folgen. Als Beispiel wird in der Abbildung das Basissystem des entworfenen finalen Prototyps gezeigt. Dieses hat eine relativ fixierte Informationsverarbeitung und nähert sich einer produzierbaren Lösung.

Weiterhin ist die Werkzeugunterstützung zur teil- bzw. vollautomatischen Betrachtung und Auswahl von interessierenden Eigenschaften sinnvoll. In dem Forschungsprojekt von Herrn Paulsami [Pau14] wurde dafür ansatzweise die Nutzung der Entwicklungsumgebung LabVIEW mit dem integrierten NI Statechart Module untersucht und an einem Beispiel erprobt. Hierzu wurden die unterschiedlichen Wagentypen mit deren Merkmalen durch Statecharts in LabVIEW beschrieben. Diese Statecharts besitzen eine Verknüpfung mit in LabVIEW entwickelten Modellen, die zur Realisierung dieser Merkmale notwendig sind.

In der Masterarbeit von Herrn Kirchhoff [Kir14] wurde das Thema der Produktlinie bei der Entwicklung eines Eingebetteten Systems für diese Dissertation untersucht. Basierend auf zwei grundlegenden Prinzipien des Produktlinienentwurfs wurden die Ansätze zur Betrachtung von unterschiedlichen Softcore-Varianten als Bestandteil von Produktlinienlösungen dargestellt. Am Bespiel des Konzeptes ViSARD wurde ein Feature Diagramm entworfen, das die Variabilität und die Grundfunktionalität des Softcore-Prozessors innerhalb einer Verwendung in der Produktlinie beschreibt (s. Anhang A.10). Hier sind enger gefasste Merkmale der Konfiguration und der Arithmetik wie z.B. Grundfunktionen, trigonometrische Funktionen und Konvertierungsfunktionen implementiert.

Bei den Untersuchungen hat sich gezeigt, dass es gerade bei Softcores durch die Verwendung der Produktlinientechnologie in abzuleitenden produzierbaren Basissystemen zur deutlichen Ressourcenreduktion bei geringem Entwicklungsaufwand kommt.

ZEfIRA unterstützt unter anderem das frühe modellbasierte Testen und die Ableitung von Testfällen während des Entwurfs für die spätere Inbetriebnahme. Nach [Fa10] ist es sinnvoll, bei dem Domänenentwurf der Produktlinie auch die Tests und Testfälle für die späteren abgeleiteten Applikationsvarianten zu entwickeln. Das lässt sich auch günstig in die Problematik der Basissystemableitung integrieren. Im finalen Prototyp entsprechend dem Abschnitt 5.2 konnten die Tests des erweiterten Prototyps, die auf seine Konfiguration zutrafen, übernommen werden.

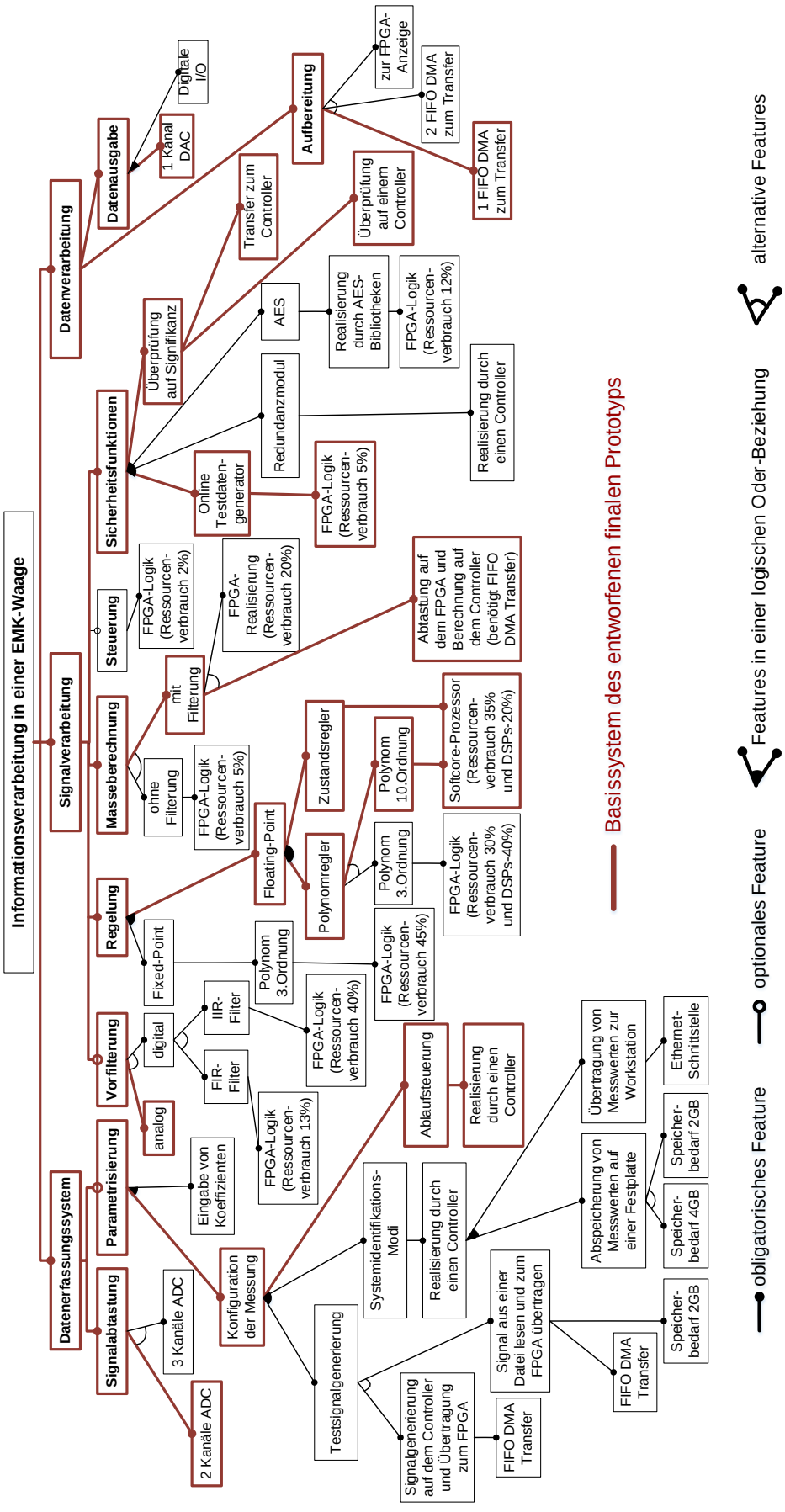


Abbildung 6.1: Feature Diagramm unterschiedlicher Basissysteme

7 Zusammenfassung und Ausblick

Die steigende Komplexität von Eingebetteten Systemen und das damit verbundene wissenschaftliche Interesse besonders beim Einsatz in der Mess- und Automatisierungstechnik sind Gegenstand der vorliegenden Dissertation. Bei der Entwicklung von Systemen in diesen Gebieten befasst sich die Forschung mit Prinzipien, Methoden und Techniken zum Software-, Hardware- und Hardware-Software-Co-Entwurf. Für die Beherrschung des Entwurfs sind modellbasierte Methoden erforderlich. Diese erlauben die formale, zumeist hierarchische, Beschreibung und die frühe Validierung und Verifikation sowie die Optimierung der Plattformstruktur und der Funktionsverteilung. Eine wichtige Rolle spielt die Vorgehensweise in einem definierten Prozess, um bei der Entwicklung die systematische Organisation von Arbeitsabläufen zu realisieren und dabei unter anderem die Fehlervermeidung und Maßnahmen für die möglichst frühzeitige Fehlererkennung und –beseitigung zu fördern. Die wichtigsten Ziele neben der effektiven Entwicklung sind Qualitätserhöhung, Kostenreduzierung und Nachvollzierbarkeit.

Die in der Arbeit betrachtete messtechnische Domäne besitzt Anwendungsbereiche mit außergewöhnlichen Anforderungen an die Informationsverarbeitung. Im messtechnischen Teilgebiet „dynamische Wägetechnik“ sind die Lösungen dadurch gekennzeichnet, dass mit sehr hohen Auflösungen und kleinen Messunsicherheiten bei schnellen Messungen in einem mechanisch gestörten Umfeld gearbeitet wird. Diese Geräte erfordern komplexe und hochleistungsfähige Informationsverarbeitungsfunktionen zur Erzeugung der Messergebnisse. Vor allem geht es dabei um Signalverarbeitung (z.B. digitale Filter, Regelalgorithmen und Störunterdrückung). Für die Implementierung der dafür verwendeten Algorithmen werden Eingebettete Systeme verwendet, die eine entsprechende Reaktionszeit, Rechen- und Kommunikationsleistung realisieren. Diese Systeme werden zunehmend mit rekonfigurierbarer Hardware in Form von FPGAs realisiert. Sie erlauben eine modulare, flexible und leistungsfähige Systemkomposition. Als Besonderheiten in der Messtechnik und insbesondere der Wägetechnik treten Anforderungen an die Eichfähigkeit und Metrologische Sicherheit auf. Ein für die Entwicklung genutzter Prozess soll zertifizierbar sein und nachweisen, dass die Metrologische Sicherheit im entworfenen System realisiert werden kann und die Forderungen von Eichbehörden und anderen gesetzlichen Einrichtungen erfüllt werden. Weiterhin trägt ein weitgehend formalisierter Prozess dazu bei, dass das spätere Produkt deutlich weniger Entwicklungsfehler aufweist. Schon aus ökonomischen Gesichtspunkten wird deshalb die Anwendung eines derartigen Prozesses gefordert.

Im Rahmen des Standes der Technik wurden im Kapitel 2 die benötigten Grundlagen des zugrundeliegenden Themenbereiches betrachtet. Beginnend mit den allgemeinen Grundbegriffen und Definitionen aus dem messtechnischen Gebiet wurden die bestehenden Prinzipien der digitalen Informationsverarbeitung einschließlich spezifischer gesetzlicher

und Sicherheitsanforderungen dargestellt. Anschließend wurde auf die unterschiedlichen Arten von bekannten Vorgehensweisen zur Entwicklung von Eingebetteten Systemen eingegangen. Darauf folgte eine Diskussion bezüglich deren Vor- und Nachteilen. Weiterhin wurden Entwicklungsumgebungen analysiert und vorgestellt, die nicht nur einen modellbasierten sondern auch plattformspezifischen Entwurf ermöglichen. Es wurde nachgewiesen, dass sich ein durchgängiger modellbasierter Entwurf bisher noch nicht in der Breite durchgesetzt hat. Oft ist es nötig, dass mehrere Modellierungs- und Entwicklungswerkzeuge parallel zum Einsatz kommen (z.B. Mechanik-, Software-, Hardware- und Algorithmenmodelle). Es gibt bislang keinen geschlossenen professionellen Modellierungsansatz, der alle Entwicklungsphasen für unterschiedliche Teildisziplinen effizient umsetzt. Bezogen auf den plattformspezifischen Entwurf wurden im nächsten Teilabschnitt die Grundlagen zu verschiedenen Implementierungsarten einschließlich einer detaillierteren Betrachtung der Besonderheiten und Vorteile FPGA-basierter Systeme dargestellt. Ein weiterer Abschnitt behandelt Validierungs- und Verifikationstechniken sowie das Testen und die Integration in den Entwicklungsprozess in allen Entwurfsphasen.

7.1 Forschungsergebnisse

Kern dieser Arbeit ist die Erarbeitung und Formulierung eines zertifizierbaren Prozesses (ZEfIRA) für die Domäne „Informationsverarbeitung in messtechnischen Systemen“, der anschaulich als 3W-Prozess bezeichnet wird. Dieser stellt eine spezielle Vorgehensweise für die Entwicklung von Eingebetteten Systemen dar, wobei Metrologische Sicherheit, Eichfähigkeit und deshalb insbesondere auch Validier- und Verifizierbarkeit als Kriterien im gesamten Entwurfsprozess berücksichtigt werden. Die Zielplattformen können dabei heterogen sein (Software, rekonfigurierbare Hardware, herkömmliche Hardware), wobei die Partitionen variabel sind und der funktionale Entwurf für die einzelnen Partitionen gemeinsam erfolgt. ZEfIRA ist evolutionär angelegt und führt die Analyse von einem eventuellen Vorläufersystem über eine modellbasierte prototypische Entwicklung bis zu einer produzierbaren Lösung (Produkt) durch. Das Vorläufersystem (erster W-Prozess) übernimmt die Rolle eines Mustersystems, das prinzipiell ähnliche Aufgaben realisiert, die für das folgende, aktuell zu entwickelte Produkt modifiziert, ergänzt und zumeist mit anderen Parametern versehen werden. Die Dissertation bezieht sich schwerpunktmäßig auf die Entwicklung des Prototyps (zweiter W-Prozess). Die Prototypenentwicklung benutzt als Zielplattformen leistungsfähige und dadurch auch kostenintensive FPGA-basierte Eingebettete Systeme. Dieses ermöglicht, verschiedene Lösungsvarianten für die funktionalen Eigenschaften und deren weitgehend optimale Realisierung flexibel zu untersuchen und genauere Spezifikationen von unklaren Anforderungen sowie die Demonstration und Validierung von Entwürfen durchzuführen. Dabei spielt der Aspekt „kostenintensiv“ auf Grund des Charakters eines Prototyps zumeist eine untergeordnete

Rolle. Eine weitere Aufgabe des Prototyps ist die Systemidentifikation und Analyse des Einbettenden Systems. Der dritte W-Prozess in ZEfIRA ist die Produktentwicklung unter Verwendung von anwendungsspezifischen Zielplattformen für die Produktion in relevanten Stückzahlen. Dabei soll auf Basis der mit dem Prototyp gefundenen Realisierung ein Produkt unter Berücksichtigung der endgültigen technischen, technologischen und wirtschaftlichen Parameter als weitgehend optimale produzierbare Lösung entstehen. Um einen effektiven Ablauf in ZEfIRA zu gestalten und die gegenseitigen Einflüsse auf die gleiche Art und Weise zu behandeln, werden die Teilprozesse mittels modifizierter W-Modelle beschrieben. Somit läuft der Prozess ZEfIRA nach dem entwickelten 3W-Modell ab. Eine weitere Besonderheit des Konzepts ist die in der Dissertation genau definierte Metrologische Sicherheit. Darunter wird die Gewährleistung der korrekten Funktion in dem Informationsverarbeitungssystem unter Berücksichtigung der geforderten messtechnischen und gesetzlichen Anforderungen verstanden. Der Nachweis der Metrologischen Sicherheit wird entsprechend ZEfIRA in allen Entwurfsetappen, beginnend in den höheren Abstraktionsebenen und systematisch weiter in den niedrigeren, geführt.

Im Kapitel 4 wurden als Schwerpunkt die Prozessschritte der effektiven Prototypenentwicklung auf Basis von ZEfIRA genauer beschrieben. Ausgehend von der werkzeugunterstützten Anforderungsanalyse bis zur Implementierung mit dazugehörigen Verifikations- und Validierungsaktivitäten wird das Eingebettete System (Informationsverarbeitungssystem) abhängig von den Forderungen des Einbettenden Systems (Messsystem) modellbasiert entwickelt. Die Entwicklung von Methoden, Prinzipien und Techniken zur Unterstützung eines durchgängigen Entwurfes war Gegenstand dieser Arbeit.

Für die Anforderungsanalyse wurde die zielbasierte Anforderungsermittlung gewählt. Dabei sind sowohl die speziellen Anforderungen an die Metrologische Sicherheit und die Eichfähigkeit als auch frühzeitig die Anforderungen der Produktentwicklung zu berücksichtigen. Beim funktionalen Entwurf wurden die Prinzipien zur Beschreibung des statischen und dynamischen Verhaltens der Systemkomponenten sowie die Techniken der strukturellen Verfeinerung betrachtet. Zur Verfeinerung der funktionalen Spezifikation wurde beispielweise ein Modul zur Analyse von Analog-Digital-Wandlern in MATLAB/Simulink entwickelt.

Im technischen modellbasierten Systementwurf werden grundsätzlich Modelle des Eingebetteten Systems als Ergänzung entworfen, wobei eine gemeinsame funktionelle Betrachtung zusammen mit der Informationsverarbeitung möglich wird. Entsprechend ZEfIRA muss eine teilweise Übernahme von modifizierten Modellen aus dem Vorläufersystem erfolgen. Diese Schritte wurden anhand von Beispielen aus dem bearbeiteten konkreten Projekt betrachtet. Die weiteren entwickelten Verfahren beziehen sich auf die plattformunabhängige Datentyp-, Fehler- und Zeitanalyse sowie auf einige Funktionen

zur Gewährleistung der Metrologischen Sicherheit. Es wurde das Prinzip der wiederverwendbaren Modelle realisiert, das den Prozess ZEfIRA für weitere Generationen von Prototypen und Produkten unterstützt (z.B. Modelle der Komponenten des Einbettenden Systems, Bibliotheken von Schnittstellen, Modelle zur Signalverarbeitung und zur Fehleranalyse von mathematischen Operationen).

Weiterhin beschreibt die Arbeit den verfeinerten technischen Entwurf speziell für rekonfigurierbare Hardware. Ein wichtiger Aspekt ist die Umsetzbarkeit eines FPGA-Entwurfs unter Latenzschranken und Ressourcenbeschränkungen. Für die effektive und komplexe Informationsverarbeitung mittels FPGA-Plattformen wurde die Anwendung wiederverwendbarer Komponenten in ZEfIRA integriert. Exemplarisch wurden eine Bibliothek zur Berechnung von Fließkomma-Arithmetik implementiert und speziell entworfene Varianten einer Softcore-Prozessorfamilie (LiSARD in LabVIEW und ViSARD in VHDL) einschließlich deren Codegeneratoren beschrieben. Zur plattformspezifischen Ressourcen- und Zeitanalyse wurden unter anderem formale Methoden mittels speziell entwickelter Petri-Netze integriert. Dafür war es notwendig, die Petri-Netze geeignet zu interpretieren. Bei der nachfolgenden FPGA-Implementierung wurden Arbeiten zum „Modulentwurf“ anhand von Softcore-Prozessoren einschließend deren Programmen und direkten FPGA-Umsetzungen (Logik und IP-Cores) durchgeführt. Für die Realisierung eines Softcore-Programms wurden unterschiedliche Wege der Codeerzeugung betrachtet und Ansätze der automatischen Codeerzeugung (über C-Code oder auf Basis einer MATLAB/Simulink-Bibliothek) entwickelt. Zum Testen von Softcore-Programmen wurde das Testsystem LiCT realisiert, das zur Syntax- und Funktionalitätsprüfung benutzt werden kann. Abschließend wurde das Testen der realisierten und implementierten Systems auf den behandelten Abstraktionsebenen betrachtet.

Als Demonstrationsbeispiel für den Nachweis der Anwendbarkeit von ZEfIRA inklusive der erarbeiteten Methoden und Prinzipien diente die prototypische Entwicklung des Informationsverarbeitungssystems in einer EMK-Waage. Ziele der Entwicklung einer verbesserten Generation anhand des neuen Prototyps war eine Leistungssteigerung bezogen auf die Verringerung der Messunsicherheit, die Erhöhung der Auflösung und die deutliche Verkürzung der Messzeit im Vergleich zum vorgestellten Vorläufersystem. Für die Informationsverarbeitung waren grundsätzlich zwei Teile ausschlaggebend: die optimal an das Gesamtsystem angepassten Signalverarbeitungs-, Regelungs- und Sicherheitsalgorithmen sowie deren zugeschnittene Implementierung. Letztere musste mit variierenden Leistungsparametern, wie z.B. Latenz, Verarbeitungskomplexität und Genauigkeit, für eine modulare eingebettete Messdatenverarbeitungsplattform mit einem eingebetteten Controller sowie mit einem FPGA realisiert werden. Ein Nebenziel war eine effektivere Entwicklung bezogen auf den personellen und Zeitaufwand. Die durchgängige modellbasierte Entwicklung und Umsetzung von Algorithmen wurde in der Entwicklungsumgebung LabVIEW mit speziellen Erweiterungen durchgeführt. Bei

exemplarischer Umsetzung von ausgewählten Schritten der Prototypenentwicklung wurden unterschiedliche Partitionierungen zwischen dem Einbettenden und dem Eingebetteten System und innerhalb der Informationsverarbeitung in verschiedenen Hierarchieebenen iterativ untersucht. In diesem Zusammenhang entstanden regelmäßig veränderliche Anforderungen an die Informationsverarbeitung auf Grund der vorhandenen Lösungsvariabilität. Bei der Anforderungsanalyse wurde das Konzept der Werkzeugunterstützung in LabVIEW (über NI Requirements Gateway) umgesetzt. Im funktionalen und technischen Entwurf wurden die entworfenen und im Kapitel 4 beispielhaft beschriebenen Modelle des Eingebetteten und des Einbettenden Systems verwendet. Das Ergebnis der Prototypenentwicklung waren drei unterschiedliche Umsetzungen: ein minimierter, ein erweiterter und ein finaler Prototyp. Der letzte ermöglicht die Weiterführung der Entwicklung des Eingebetteten Systems mit relativ fixierter Informationsverarbeitung und nähert sich damit einer produzierbaren Lösung an. In ihm wurden z.B. alle Funktionen zur Analyse des Einbettenden Systems ausgespart und bei der Umsetzung der Algorithmen auf Ressourcensparsamkeit geachtet.

Die Verwendung von FPGA-basierten Plattformen ermöglichte die Untersuchung und Realisierung von komplexen Algorithmen mit kürzen Latenzen (im Projekt konnten verschiedene Regelungskonzepte bei unterschiedlicher Abtastfrequenz bis zu 300 kHz implementiert werden). Es wurde ein großer Flexibilitäts- und Wiederverwendungsgrad bei der Prototypenentwicklung geschaffen und nachgewiesen, so dass die implementierten Lösungen leicht an unterschiedliche messtechnische Systeme angepasst werden können.

Im Abschnitt 5.3 wurde eine Einschätzung der Effektivität von ZEfIRA, ausgehend von den im Projekt durchgeführten Arbeiten, gegeben. Es wurde erkennbar, dass die Verkürzung der Entwicklungszeit als entscheidender Vorteil des ZEfIRA-Prozesses nicht bei der ersten Produktiteration entsteht. Man kann aber anhand der Qualität, Wiederverwendbarkeit und Variabilität der Lösungen extrapolieren, dass er bei der Entwicklung von weiteren Generationen von Prototypen und damit implizit auch von Produkten entstehen wird.

Abschließend wurde der in der Dissertation ansatzweise bearbeitete Aspekt zur Erweiterung des Konzeptes ZEfIRA für den Produktlinienentwurf dargestellt. Basis sind Feature Diagramme. Mittels eines konkreten Diagramms wurden unterschiedliche Gestaltungen mehrerer im Projekt untersuchter Basissysteme beschrieben. Dieses Diagramm kann im Weiteren benutzt werden, um aus der Produktlinie eine Anwendung (Erzeugnis mit spezifizierten ausgewählten Merkmalen) abzuleiten. Als Teillösung wurde die Produktlinie Softcore untersucht und entwickelt.

7.2 Zukünftige Arbeiten

Ausgehend von den für die Dissertation erarbeiteten und dargestellten Ergebnissen lassen sich die weiterführenden Arbeiten in die folgenden Schwerpunkte einordnen:

1. *Weiterentwicklung von Methoden, Techniken und Prinzipien für die Unterstützung und Effektivierung von Prozessschritten in ZEfIRA gemäß des fortschreitenden Standes der Technik*

Dieses bezieht sich vor allem auf die Verwendung von formalen Entwurfs- und Testmethoden, da diese eine nachvollziehbare Entwicklung einschließlich Verifikations- und Validierungsaktivitäten ermöglichen. Interessant wären Betrachtungen zur iterativen Anforderungsanalyse. Für den hierarchischen Entwurf sollten eigenschaftserhaltende Kompositionstechniken behandelt werden. In diesem Zusammenhang würden Methoden zur Traceability von Bedeutung sein. Beim Übergang zur Zielplattform ist eine stärkere Integration von Wiederverwendungstechniken sinnvoll. Zum Teilgebiet der spezialisierten Softcore-Prozessoren ist es erstrebenswert, Untersuchungen zu applikationsspezifischen Befehlssatzarchitekturen (ASIPs) durchzuführen.

Weiterhin ist das Gebiet System-on-Chip von Interesse. Die steigende Systemkomplexität mit höherer Integrationsdichte bei der eingebetteten Hardware wird in Zukunft dadurch gekennzeichnet, dass noch komplexere Informationsverarbeitungssysteme auf einem Chip untergebracht werden können. Dieses ermöglicht die Entwicklung von kleinen wiederverwendbaren Teilsystemen, die für unterschiedliche Ansätze in der Messtechnik verwendet werden könnten. Außerdem ist eine Leistungssteigerung bei einer Verringerung der Kosten zu erwarten. Für die effektive modulare Kommunikation sind dabei Network-on-Chip-Lösungen (NoC) zu betrachten.

Auf dem Gebiet der Validierung und Verifikation ist die Entwicklung von geeigneten Testumgebungen in den Entwurfsprozess zu integrieren. Für die logisch-zeitliche Verifikation sollte die werkzeugunterstützte Verwendung von erweiterten Petri-Netzen und Discrete-Event-Modellen detaillierter betrachtet werden. Insbesondere erscheinen Untersuchungen zur Extraktion derartiger Modelle aus den Datenflussmodellen, wie sie z.B. MATLAB/Simulink oder LabVIEW benutzen, naheliegend. Für das Testen von Implementierungen von komplexen Algorithmen auf FPGAs sind Methoden zum White-Box-Test zu betrachten und zu unterstützen. Diese würden über den heute häufig eingesetzten Black-Box-Test deutlich hinausgehende Ergebnisse liefern. Als Beispiel ist dabei die Codeanalyse von Bedeutung.

2. Weiterführende Untersuchungen zu einer für die messtechnische Anwendung offenen erweiterbaren Entwicklungsumgebung

Die Untersuchungen zeigten, dass unterschiedliche Messsysteme ähnliche Anforderungen bei der Entwicklung besitzen. Für zukünftiges effektives Arbeiten mit ZEfIRA ist es sinnvoll, eine speziell dafür geeignete Entwicklungsumgebung zu realisieren. Diese soll eine durchgängige Entwicklung, beginnend mit der Anforderungsanalyse über den Entwurf einschließlich Verifikations- und Validierungsaktivitäten bis hin zum Testen und zur Inbetriebnahme werkzeugbasiert ermöglichen. Durch die Eigenschaften „offen und

erweiterbar“ wäre es möglich verschiedene Tools zu verbinden, was den Zugang zu verschiedenen Abstraktionsebenen, Modellen und Methoden anhand eines einheitlichen Vorgehens (ZEfIRA) schaffen kann. Aus der Sicht der Informationsverarbeitung und insbesondere der Realisierung über FPGAs ist der Zusammenhang von gesetzlichen Regelungen und weitergehenden Forderungen zur Metrologischen Sicherheit und Eichfähigkeit in Vorschriften, Empfehlungen und Richtlinien mit zu gestalten. Ein weiterer interessanter Aspekt wäre die Betrachtung von redundanten Strukturen in der Informationsverarbeitung zur Erhöhung der Metrologischen Sicherheit.

3. *Nachweis der Anwendbarkeit von ZEfIRA vom Prototyp bis zum Produkt*

In der Dissertation konnte die Anwendbarkeit von ZEfIRA nur auf Basis der Entwicklung eines neuen Prototyps am Beispiel genau einer Wägezelle untersucht werden. Damit ist natürlich die Effektivität des Vorgehens ZEfIRA und der konkret weiterentwickelten Entwurfsschritte nur in Teilen praktisch nachgewiesen worden. In weiteren Arbeiten sollte der vollständige Ablauf aller Teilprozesse (vom Vorläufersystem bis zur produzierbaren Lösung) durchgeführt und die Effektivität eingeschätzt werden.

4. *Erweiterung von ZEfIRA zum Produktlinienentwurf*

Eine weitere Forschung kann sich auf Untersuchungen im Bereich des Produktlinienentwurfes für messtechnische Anwendungen fokussieren. Dabei wäre es von Interesse, für die Anwendungsdomäne „Kraftmess- und Wägetechnik“ einen Feature-Katalog zu realisieren. Auch hier sind effektive Test- und Verifikationsstrategien zu entwickeln und zu integrieren.

5. *Weiterentwicklung des Einbettenden und des Eingebetteten Systems im Projekt der betrachteten EMK-Waage*

Die im Projekt durchgeführten Untersuchungen haben gezeigt, dass die Verbesserung der gesamten Messqualität vor allem mit Optimierungen im Einbettenden und Eingebetteten System im Verbund möglich ist. Es wäre darum interessant eine einheitliche Modellierungsmethodik für Einbettende und Eingebettete Systeme, unter anderem in der Kombination verschiedener Modelle, zu untersuchen und zu entwickeln. Zur Verbesserung der Signalverarbeitung könnten neue Regelungskonzepte (z.B. adaptive Regler) sowie neue Filteralgorithmen implementiert werden, die einen signifikanten Einfluss auf die Messgenauigkeit und Messgeschwindigkeit haben. Deren Effekt tritt aber erst dann in Erscheinung, wenn das dynamische Verhalten des Messsystems, hier der Wägezelle, daran angepasst ist

A. Anhang

A.1. Funktionaler Systementwurf

Realisierung des Elliptischen Filters in MATLAB/Simulink

```
%%% globale Parameter %%%
% Abtastfrequenz [Hz]
f = 280000;
Ts = 1/f;
%%% Festlegen der Entwurfsparameter %%%
% Verstaerkung Durchlassbereich [dB]
Rp = 0.01;
% Verstaerkung gedämpfter Bereich [-dB]
Rs = 50;
% Grenzfrequenz des Lowpassfilters [Hz]
omega_stopp = 10000;
% Bereite Übergangsbereich [Hz]
transfer = 100;
%%% Filterdesign %%%
% Definition Grenzfrequenzen Lowpass
Wp = omega_stopp/(f/2);
% Definition Übergangsbereich
Ws = (omega_stopp+transfer)/(f/2);
% Berechnung min. Filterordung
[n,Wn] = ellipord(Wp,Ws,Rp,Rs);
% Filterdesign
[z_ellip,p_ellip,k_ellip] =ellip(n,Rp,Rs,Wn);
[b_ellip,a_ellip] =ellip(n,Rp,Rs,Wn);
%Second-Order Representation
[SOS_ellip,g_ellip]=zp2sos (z_ellip,p_ellip,k_ellip);

%%% Auswertung %%%
% Amplitudengang elliptischer Filter
figure(1)
freqz(b_ellip,a_ellip,2048,f);
disp(['Filterordnung = ', num2str(n*2)])
title(['Amplitudengang von n = ' num2str(n*2) 'Elliptic Lowpass
Filter'])
% Pol-Nullstellen-Darstellung elliptischer Filter
figure (2)
phasedelay(b_ellip,a_ellip,'whole')
title(['Phasennacheilung von n = ' num2str(n*2) 'Elliptic
Lowpass Filter'])
```

Abbildung A.1: MATLAB-Programm

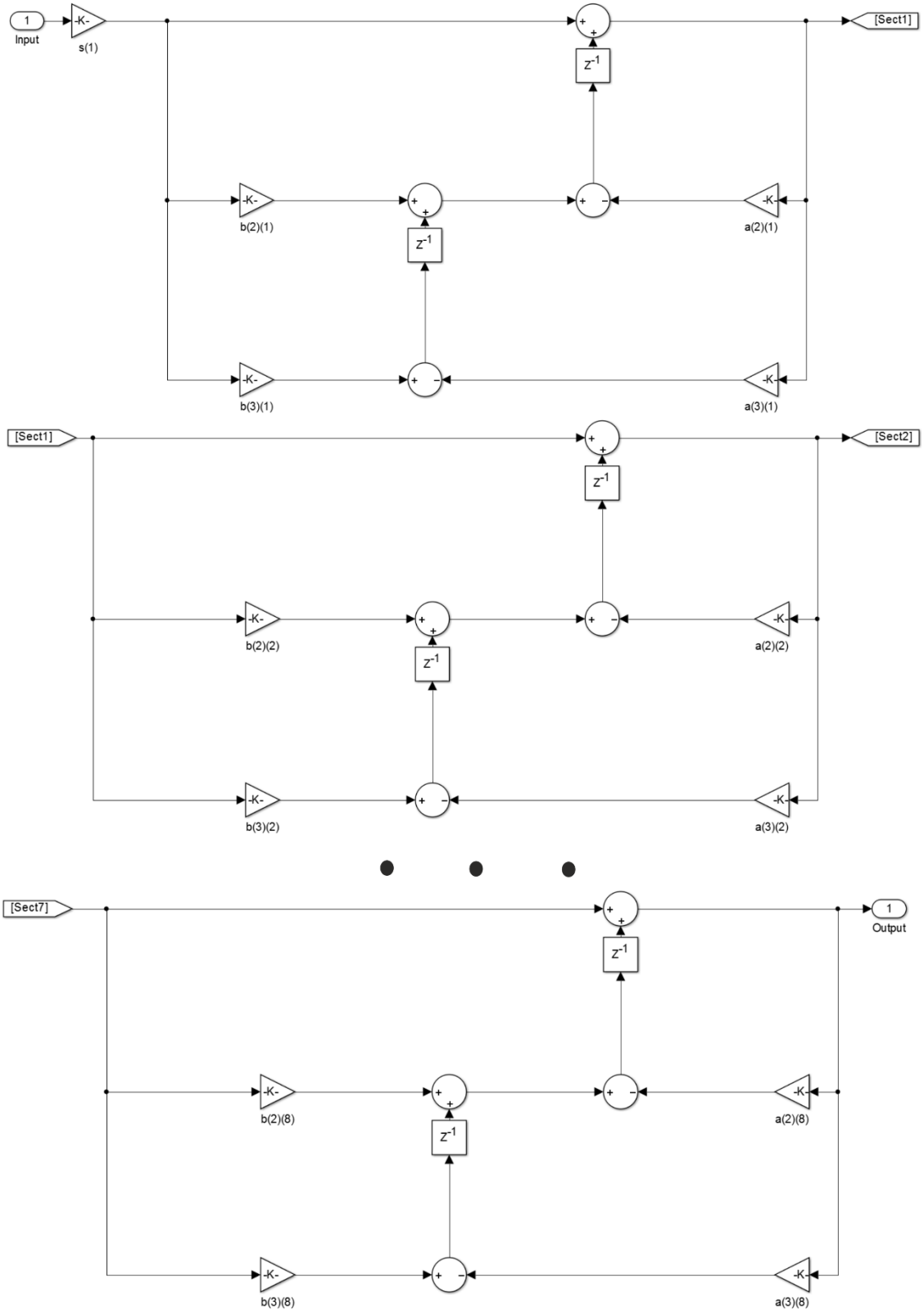


Abbildung A.2: Simulink-Modell

A.2. Technischer Systementwurf

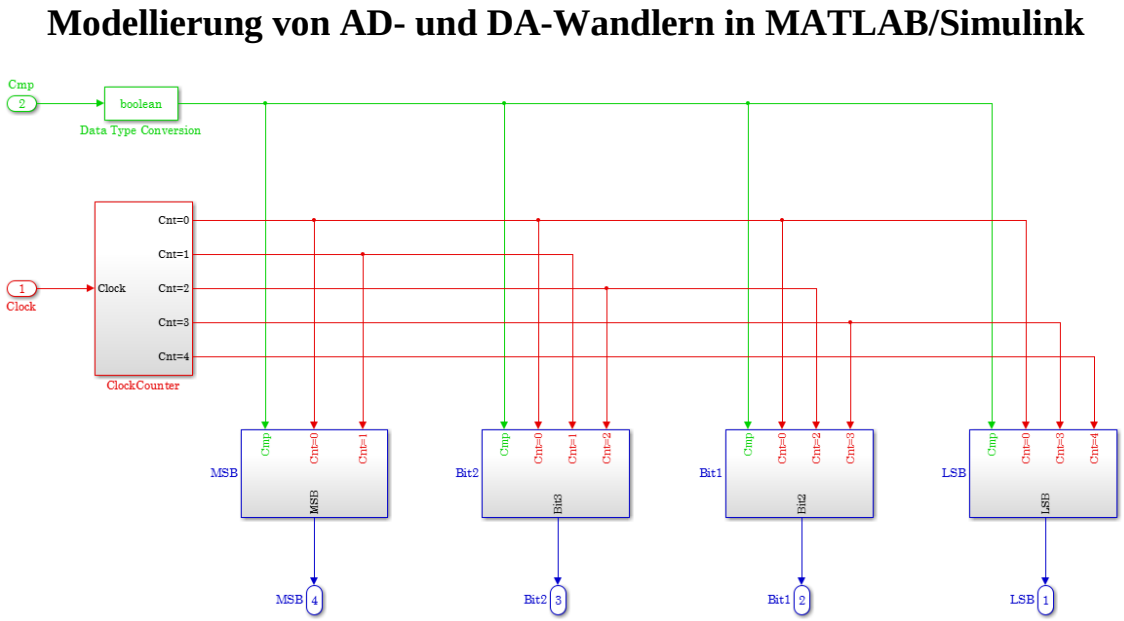


Abbildung A.3: Simulink-Modell (ADC nach dem SAR-Prinzip 4 Bit Auflösung)

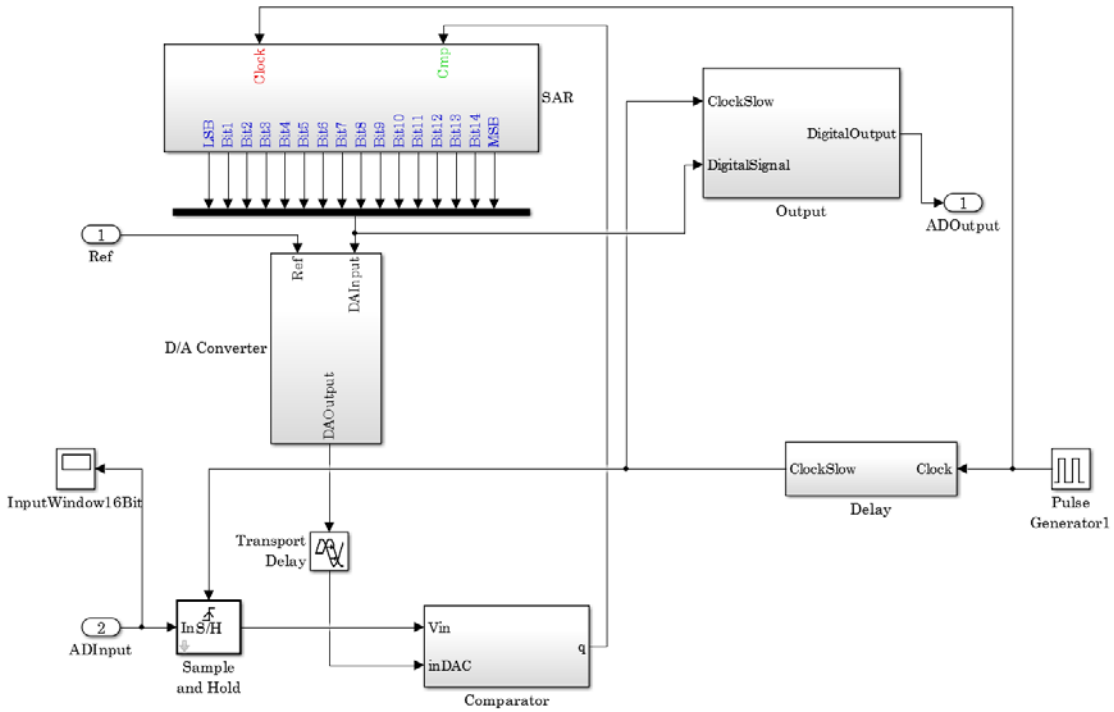


Abbildung A.4: Simulink-Gesamtmodell (AD- und DA-Wandler 16 Bit Auflösung)

Modellierung von Verschlüsselungsalgorithmen in MATLAB/Simulink

1. Realisierung des AES-Algorithmus

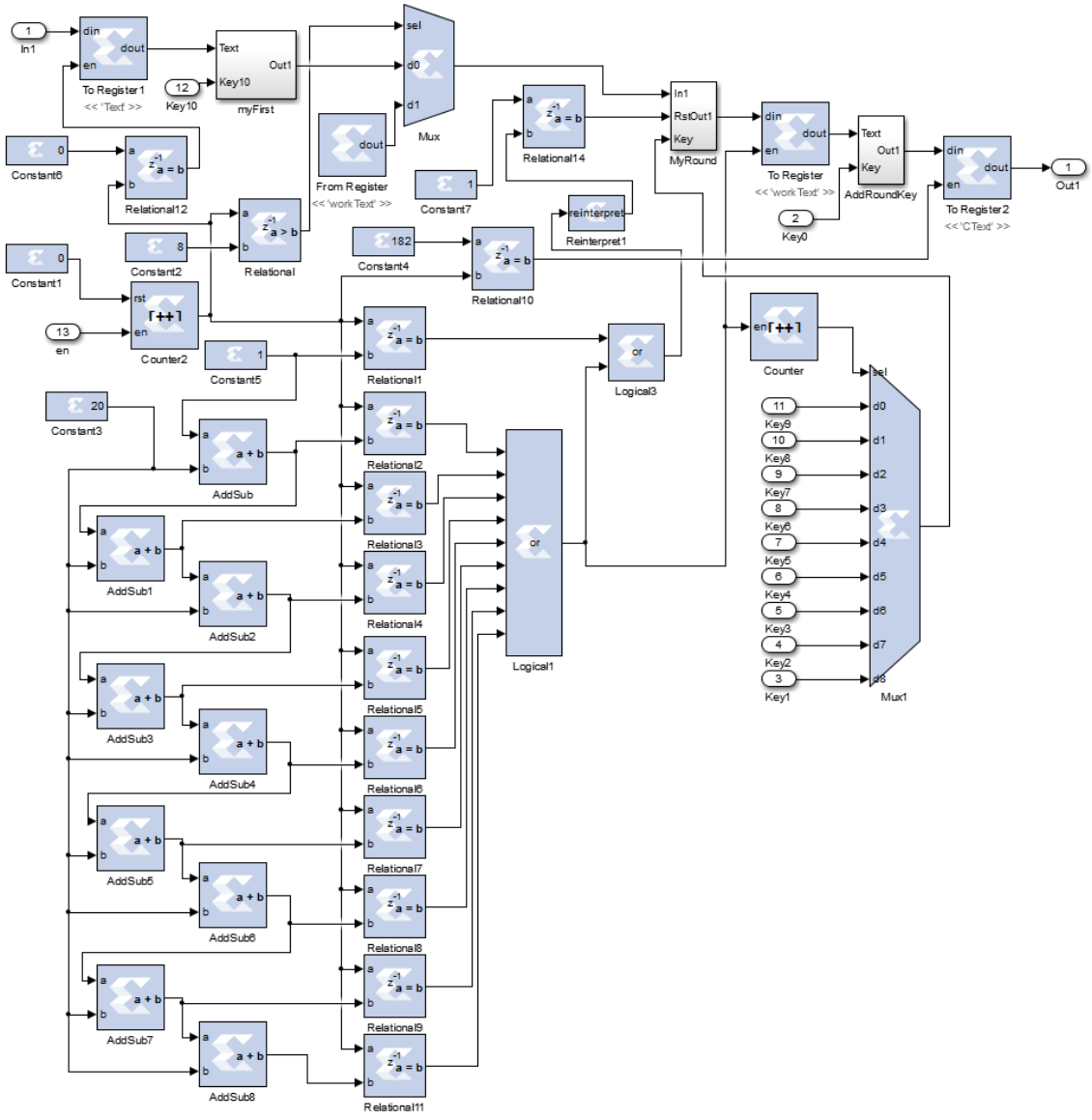


Abbildung A.5: AES-Verschlüsselung

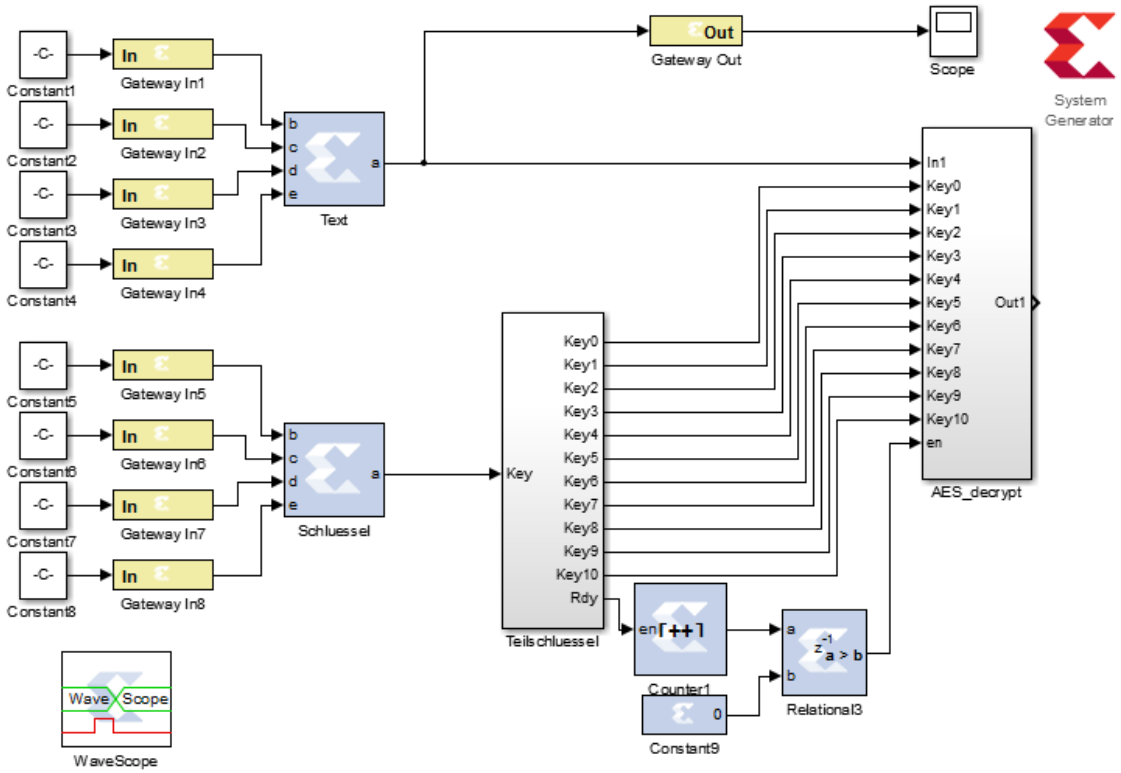


Abbildung A.6: AES-Entschlüsselung

2. Realisierung des TwoFish-Algorithmus

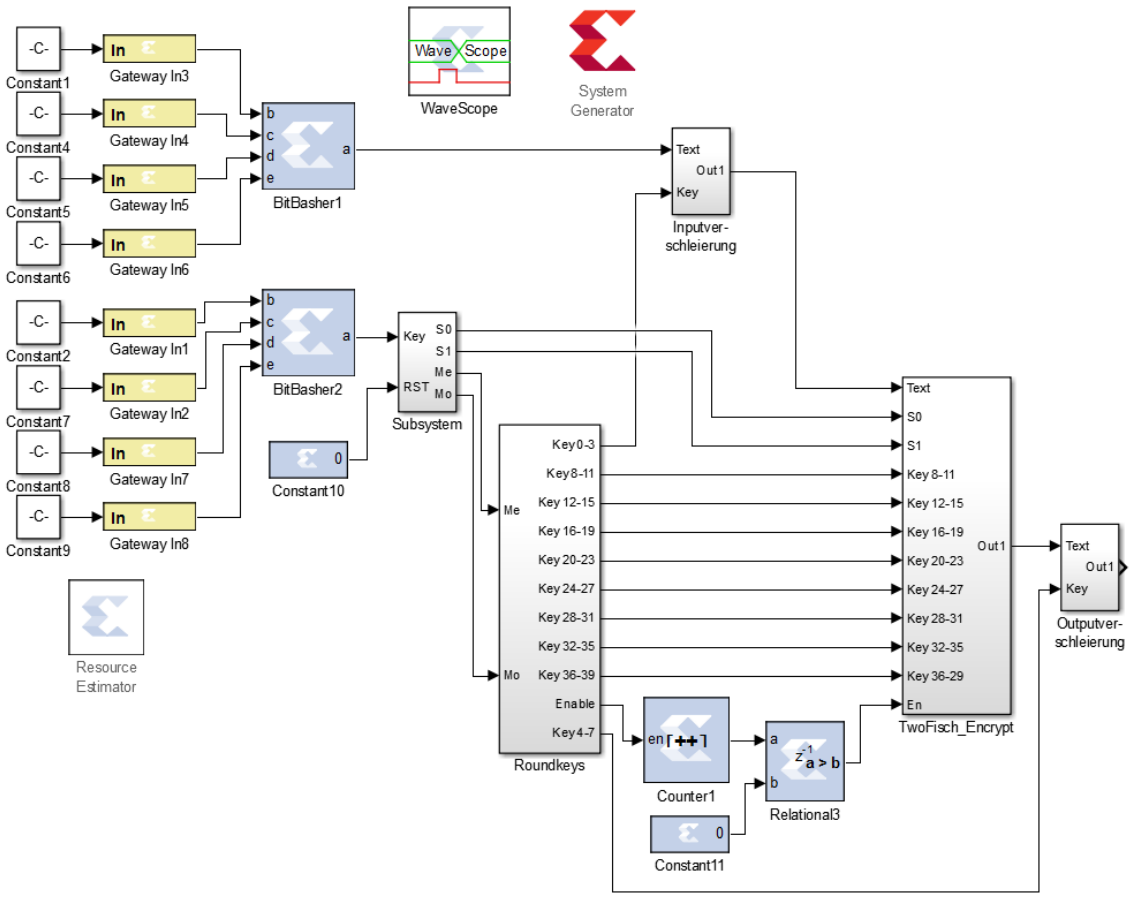


Abbildung A.7: TwoFish-Entschlüsselung

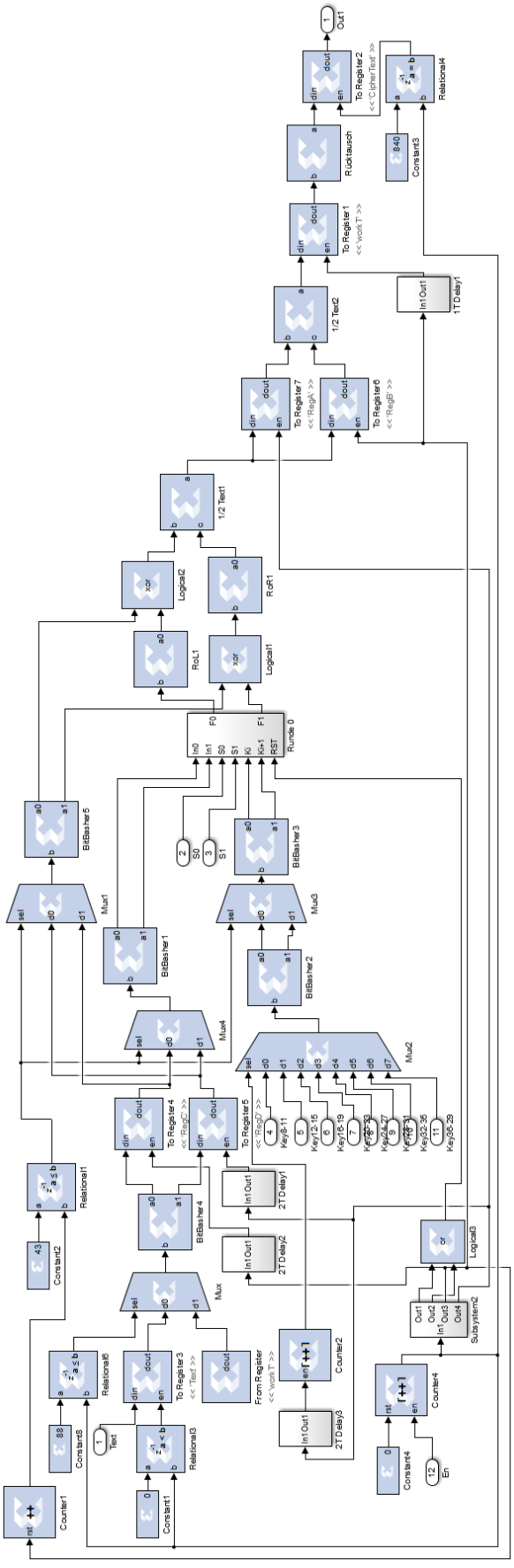


Abbildung A.8: Two-Fish-Verschlüsselung

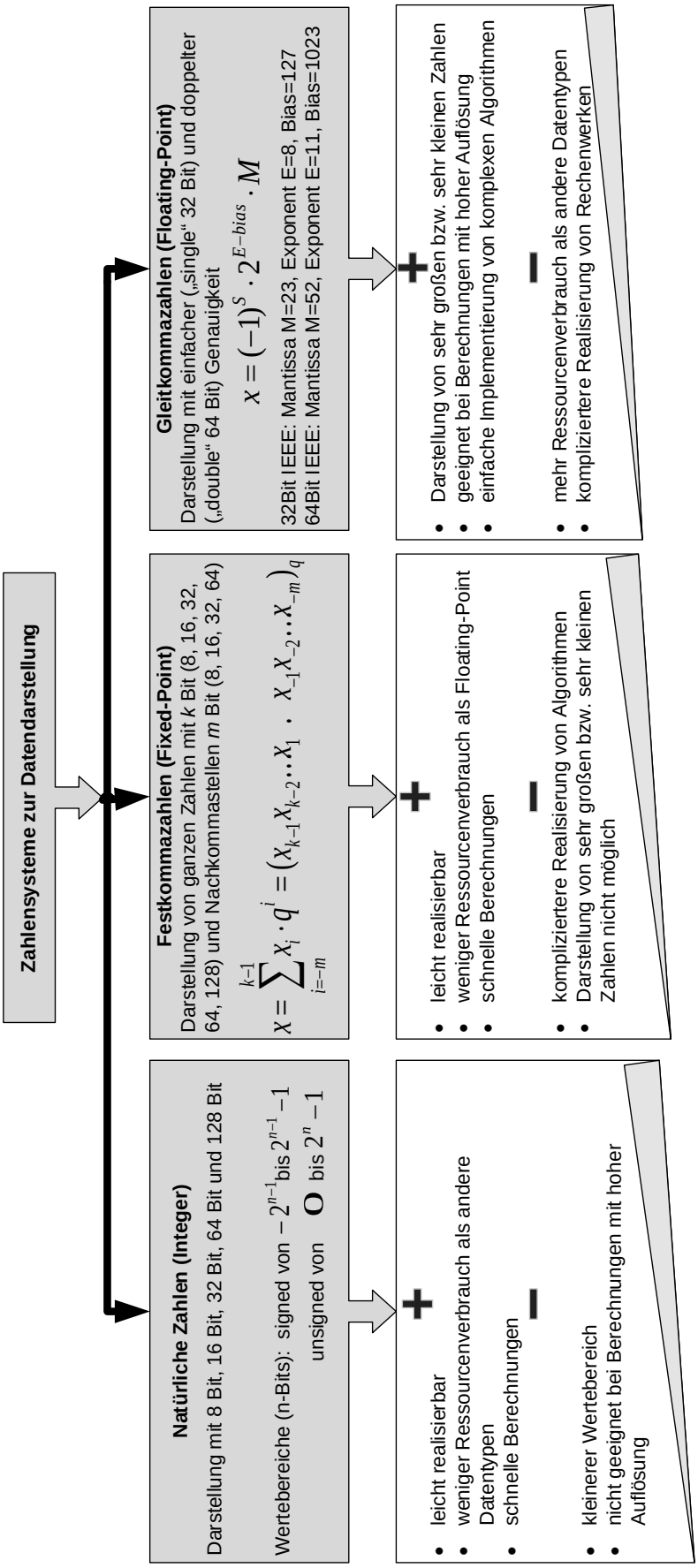


Abbildung A.9: Vor- und Nachteile von verschiedenen Datendarstellungen

A.3. Verfeinerter technischer Systementwurf

Operation		Ressourcenverbrauch					Latenz
		Total Slices	*Slice Registers*	*LUTs*	*DSP*	*RAMs*	*Ticks*
Addition	+	5,6%	3,2%	2,7%	0%	0%	5
Subtraktion	-	5,4%	3,2%	2,7%	0%	0%	5
Multiplikation	*	6,9%	5%	4,5%	0%	0%	6
Quadratwurzel	√	8,3%	6%	3,9%	0%	0%	59
Division	÷	12%	10%	5,8%	0%	0%	59
Vergleichsfunktion „gleich“	=	2,8%	1,3%	1,1%	0%	0%	4
Vergleichsfunktion „nicht gleich“	≠	2,7%	1,3%	1,0%	0%	0%	4
Vergleichsfunktion „größer“	>	2,6%	1,3%	1,1%	0%	0%	4
Vergleichsfunktion „größer gleich“	≥	2,8%	1,3%	1,2%	0%	0%	4
Vergleichsfunktion „kleiner“	<	2,7%	1,3%	1,2%	0%	0%	4
Vergleichsfunktion „kleiner gleich“	≤	2,8%	1,3%	1,2%	0%	0%	4
Konvertierung Double zu Single	**DBL/SGL**	2,8%	1,3%	1,1%	0%	0%	5
Konvertierung Double zu Integer (16 bit)	**DBL/I16**	2,9%	1,5%	1,3%	0%	0%	8
Konvertierung Integer (16 bit) zu Double	**I16/DBL**	2,8%	1,4%	1,2%	0%	0%	8

Tabelle A.1: Ressourcenverbrauch auf dem FPGA Virtex-5-LX110

Mnemonic	SGL/ DBL	Operand 1	Operand 2	Writeback Address	Writeback Command	Execution Command	Description
DQ	both	Variable Name	?	Value	none	none	Declare variable name and assign init value
In	both	Input Address	?	Variable Name	00000000	000	Mux from input address to data memory
Out	both	Variable Name	?	Output Address	00000001	000	Mux from data memory to output address
Mov	both	Variable Name	?	Variable Name	00000010	000	Op3 = Op1
Abs	both	Variable Name	?	Variable Name	00000010	100	Op3 = |Op1|
MovConst	both	Variable Name	?	Variable Name	00000010	110	Op3 = Op1, read Op1 from ConstData
Add	both	Variable Name	Variable Name	Variable Name	00000011	000	Op3 = Op1 + Op2
Sub	both	Variable Name	Variable Name	Variable Name	00000011	001	Op3 = Op1 - Op2
AbsSub	both	Variable Name	Variable Name	Variable Name	00000100	101	Op3 = |Op1|- |Op2|
Mul	both	Variable Name	Variable Name	Variable Name	00000101	000	Op3 = Op1 * Op2
Div	both	Variable Name	Variable Name	Variable Name	00000110	000	Op3 = Op1 / Op2
WbNone	both	?	?	?	00000111	000	Disables Write Enable of DRAM for 1 cycle
Sqrt	both	Variable Name	?	Variable Name	00001000	000	Op3 = sqrt(Op1)
NatExp	both	Variable Name	?	Variable Name	00011010	000	Op3 = e^{Op1}
Sin	SGL	Variable Name	?	Variable Name	00001001	000	Op3 = sin(Op1)
Cos	SGL	Variable Name	?	Variable Name	00001010	000	Op3 = cos(Op1)
SGLtoDBL	DBL	Variable Name	?	Variable Name	00001011	000	Op3 = dbl(Op1)
DBLtoSGL	DBL	Variable Name	?	Variable Name	00001111	000	Op3 = sgl(Op1)
SGLtoI16	SGL	Variable Name	?	Variable Name	00001100	000	Op3 = int16(Op1)
I16toSGL	SGL	Variable Name	?	Variable Name	00001101	000	Op3 = sgl(Op1)
DBLtoI32	DBL	Variable Name	?	Variable Name	00001100	000	Op3 = int32(Op1)
I32toDBL	DBL	Variable Name	?	Variable Name	00001101	000	Op3 = dbl(Op1)
DBLtoI16	DBL	Variable Name	?	Variable Name	00011001	000	Op3 = int16(Op1)
I16toDBL	DBL	Variable Name	?	Variable Name	00011000	000	Op3 = dbl(Op1)

Tabelle A.2: Befehlssatz des Softcore-Prozessors ViSARD (Teil 1)

Mlookup	both				00001110	000	not implemented outside ALU
MovIs Positive	both	Variable Name	Variable Name	Variable Name	00010000	000	Op3 = Op1 if sign(Op2) = 0
MovIs Negative	both	Variable Name	Variable Name	Variable Name	00010001	000	Op3 = Op1 if sign(Op2) = 1
MovIs Null	both	Variable Name	Variable Name	Variable Name	00010010	000	Op3 = Op1 if significand(Op2) = 0 and exponent(Op2) = 0
MovIsn Null	both	Variable Name	Variable Name	Variable Name	00010010	000	Op3 = Op1 if significand(Op2) != 0 and exponent(Op2) = 0
MovIs NaN	both	Variable Name	Variable Name	Variable Name	00010100	000	Op3 = Op1 if significand(Op2) != 0 and exponent(Op2) is all 0
MovIsn NaN	both	Variable Name	Variable Name	Variable Name	00010101	000	Op3 = Op1 if significand(Op2) = 0 or exponent(Op2) is not all 1
MovIs INF	both	Variable Name	Variable Name	Variable Name	00010110	000	Op3 = Op1 if significand(Op2) = 0 and exponent(Op2) is all 1
MovIsn INF	both	Variable Name	Variable Name	Variable Name	00010111	000	Op3 = Op1 if significand(Op2) != 0 or exponent(Op2) is not all 1

Tabelle A.3: Befehlssatz des Softcore-Prozessors ViSARD (Teil 2)

A.4. Anforderungsanalyse im Projekt

	A	B	C	D	E	F
1	REQ ID	LABEL	Req Text	Prio	Author	Test Requirments
2	REQ1	Display_Start_Weight	The system shall have 0.0 as the deffault start weight of the system.The system is initialised and 0.0 displayed to the user	high	abi	Validate if the weight is placed for 10 seconds
3	REQ2	Display_actual_Weight	Once the object to be measured is placed for 10 seconds the system shall process the weight information and display it to the user	high	abi	Validate if the measurement of weight is between the specific range
4	REQ3	Weight Range	The system shall measure weight ranging between 0.1mg until 220mg	high	abi	Validate if the measurement of weight is between the specific range
5	REQ4	Information Processing	The system shall use sinewave generator to get the input from the sensor and send it to the ADC	high	abi	Validate the type of FPGA
6	REQ5	Standard for Measurment Domain	The system shall be dual redundant to fulfill qualification in information system specifically for measurement domain	high	abi	Valisdate the certifiable standards
7	REQ6	Mass to Votlage Transformation	The system shall use a pressure sensor to capture the mass and gives the observed voltage output	high	abi	Validate the gram to voltage measurment
8	REQ7	Analog to Digital Conversion	The system shall use a Low-Power, Single-Channel 22-Bit Delta-Sigma ADCs with the full scale Voltage range of 10V and the digital signal will be fed to the FPGA	high	abi	Validate the sampling frequency and resolution of ADC
9	REQ8	Caliberation Requirement	The Caliberation of 22 bit ADC is provided here	high	abi	
10						

NIRG_Format_RequirementStructur | L1 Reqs | L2 Reqs | Cer Reqs | sys vs Certifable | Cer vs Sys Map

Abbildung A.10: Beschreibung von Anforderungen in Microsoft Excel

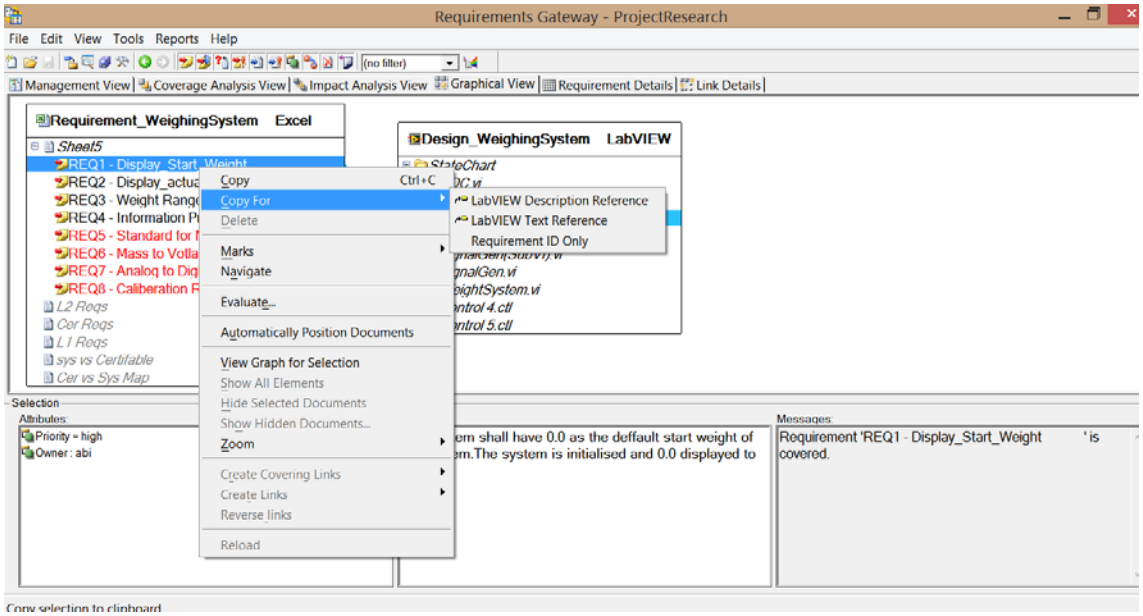


Abbildung A.11: Integration von Anforderungen in NI Requirements Gateway

A.5. Realisierung des Regelungssystems

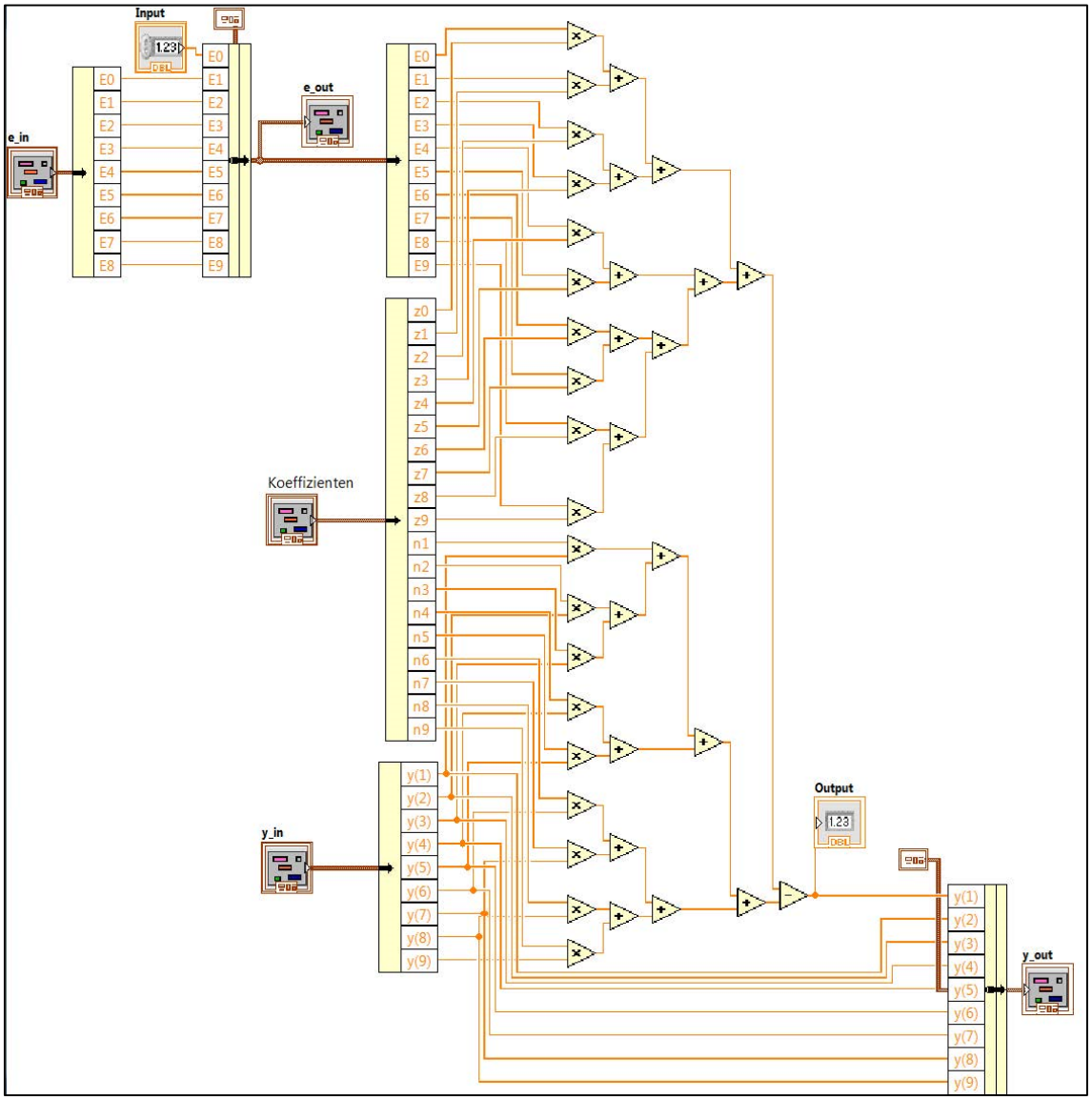


Abbildung A.12: LabVIEW-Modell des Reglers (Polynom 10. Ordnung)

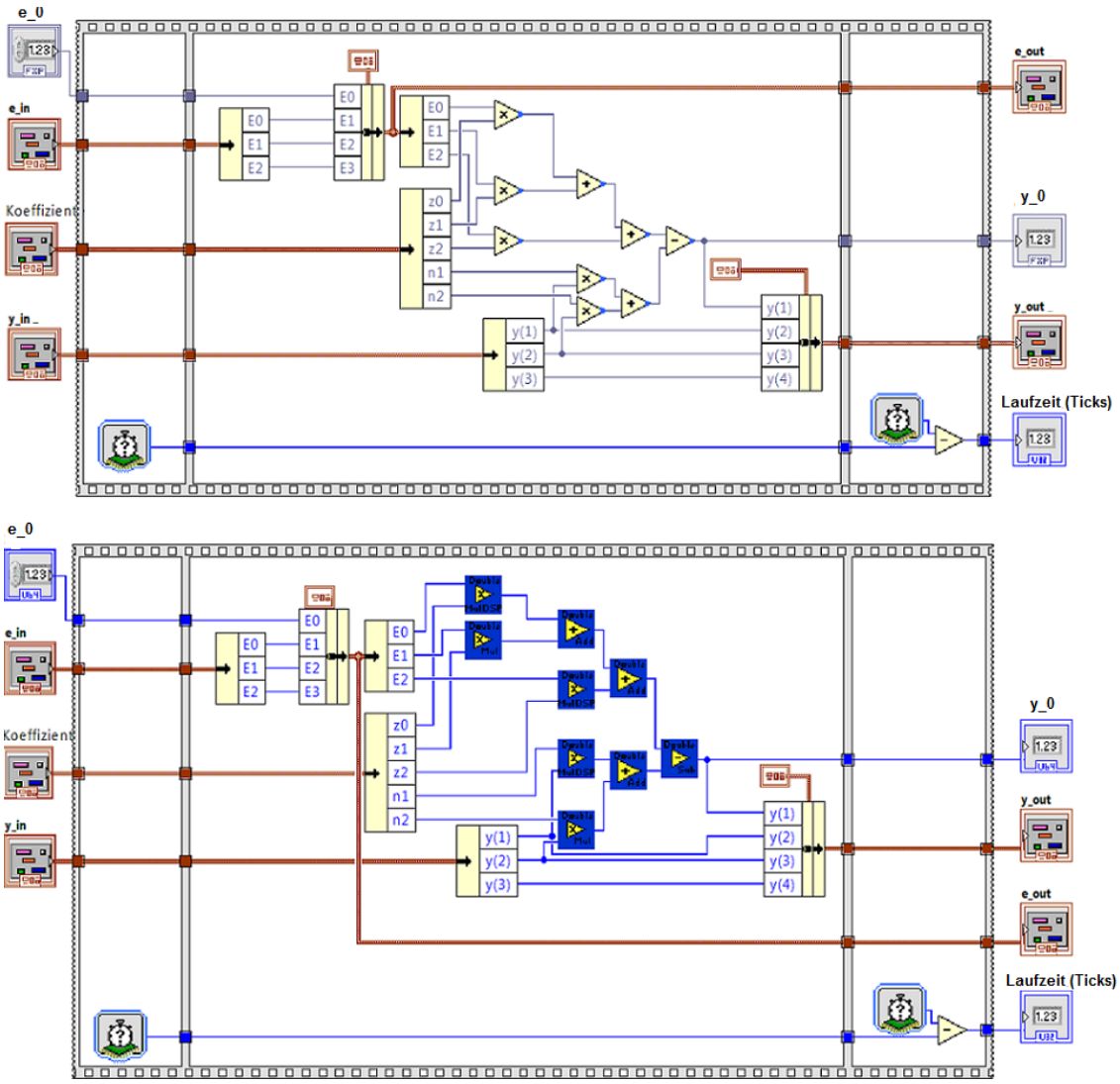


Abbildung A.13: LabVIEW-Realisierung des Polynoms 3. Ordnung

(oben: Fixed-Point, unten: Floating-Point doppelter Genauigkeit)

Softcore-Programm (Regelungssystem)

```
; Eingabewerte (Initialisierung von Ein- und Ausgängen)
dq      in_Reg      ?       0
dq      umw1        ?       0
dq      umw2        ?       0
dq      A           ?       0
dq      B           ?       0
dq      in_soll     ?       0
dq      in_e0t0     ?       0
dq      e1          ?       0
...
; Deklarierung von Eingängen e2-e8
...
dq      e9          ?       0
dq      ymin        ?       -10.0
dq      ymax        ?       10.0
dq      f           ?       0
dq      AD_umw      ?       0.00030517578
dq      DA_umw      ?       3276.8
dq      out_y0t0    ?       0
dq      y0_umw      ?       0
dq      y0_umwA     ?       0
dq      y0_umwB     ?       0
dq      y0          ?       0
...
; Deklarierung von Ausgängen y1-y8
...
dq      y9          ?       0

; Koeffizienten
dq      z0          ?       0
...
; Deklarierung von Koeffizienten z1-z8
...
dq      z9          ?       0
dq      n1          ?       0
...
; Deklarierung von Koeffizienten n2-z8
...
dq      n9          ?       0

; Lesen „Inputs“
in 0        ?       in_Reg
in 1        ?       in_soll
mul in_Reg    AD_umw  in_Reg
in 2        ?       umw1
mul in_Reg    umw1          in_Reg
in 3        ?       umw2
in 4        ?       A
```

```
in 5        ?       B

; Speicher lesen
Clookup 0 ?    z0
...
; Koeffizienten z1-z8, n0-n8
...
Clookup 18    ?    n9

; Zwischenergebnisse
dq v_M_e0z0       ?    0
...
; Reservierung von Registern
...
dq v_A_y1_9       ?    0

;Programm
sub     in_soll         in_Reg      in_e0t0

mul     in_e0t0         z0          v_M_e0z0
mul     e1              z1          v_M_e1z1
mul     e2              z2          v_M_e2z2
mul     e3              z3          v_M_e3z3
mul     e4              z4          v_M_e4z4
mul     e5              z5          v_M_e5z5
mul     e6              z6          v_M_e6z6
mul     e7              z7          v_M_e7z7
mul     e8              z8          v_M_e8z8
mul     e9              z9          v_M_e9z9
mul     y1              n1          v_M_y1n1
mul     y2              n2          v_M_y2n2
mul     y3              n3          v_M_y3n3
mul     y4              n4          v_M_y4n4
mul     y5              n5          v_M_y5n5
mul     y6              n6          v_M_y6n6
mul     y7              n7          v_M_y7n7
mul     y8              n8          v_M_y8n8
mul     y9              n9          v_M_y9n9

mov     e8              ?           e9
mov     e7              ?           e8
mov     e6              ?           e7
mov     e5              ?           e6
mov     e4              ?           e5
mov     e3              ?           e4
mov     e2              ?           e3
mov     e1              ?           e2
mov     in_e0t0         ?           e1
add     v_M_e0z0        v_M_e1z1    v_A_e0e1
add     v_M_e2z2        v_M_e3z3    v_A_e2e3
add     v_M_e4z4        v_M_e5z5    v_A_e4e5
```

```
add     v_M_e6z6    v_M_e7z7    v_A_e6e7
add     v_M_e8z8    v_M_e9z9    v_A_e8e9
add     v_A_e0e1    v_A_e2e3    v_A_e0_3
add     v_A_e6e7    v_A_e8e9    v_A_e6_9
add     v_A_e4e5    v_A_e6_9    v_A_e4_9
add     v_A_e0_3    v_A_e4_9    v_A_e0_9
add     v_M_y8n8    v_M_y9n9    v_A_y8y9
add     v_M_y6n6    v_M_y7n7    v_A_y6y7
add     v_M_y4n4    v_M_y5n5    v_A_y4y5
add     v_M_y2n2    v_M_y3n3    v_A_y2y3
add     v_M_y1n1    v_A_y2y3    v_A_y1y23
add     v_A_y6y7    v_A_y8y9    v_A_y6_9
add     v_A_y1y23   v_A_y4y5    v_A_y1_5
add     v_A_y1_5    v_A_y6_9    v_A_y1_9

sub     v_A_e0_9    v_A_y1_9    y0

mov     y8          ?           y9
mov     y7          ?           y8
mov     y6          ?           y7
mov     y5          ?           y6
mov     y4          ?           y5
mov     y3          ?           y4
mov     y2          ?           y3
mov     y1          ?           y2
mov     y0          ?           y1

mul     y0          umw2        y0_umw
mul     y0_umw      A           y0_umwA
sub     y0_umwA     B           y0_umwB
sub     y0_umwB     ymax        f
mov_ISPOS ymax      f           y0_umwB
sub     y0_umwB     ymin        f
mov_ISNEG ymin      f           y0_umwB

mul     y0_umwB     DA_umw      out_y0t0
```

; Ausgabe von Ergebnissen

```
out     y0          ?           0
out     y0_umwB     ?           1
out     out_y0t0    ?           2
```

A.6. Realisierung von Funktionen der Masseberechnung

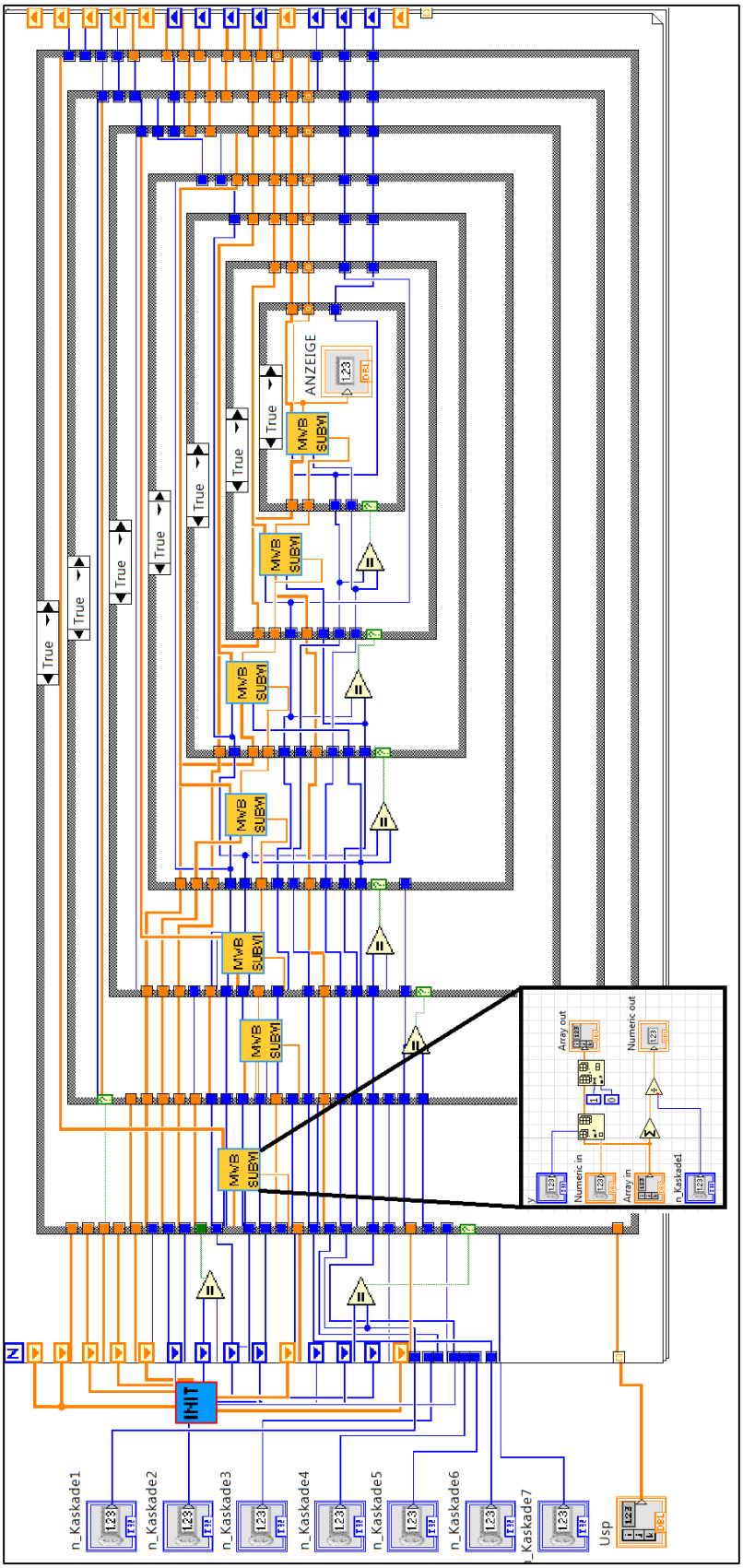


Abbildung A.14: LabVIEW-Realisierung des kaskadierten Mittelwertbildners

A.7. Realisierung von Testmodulen

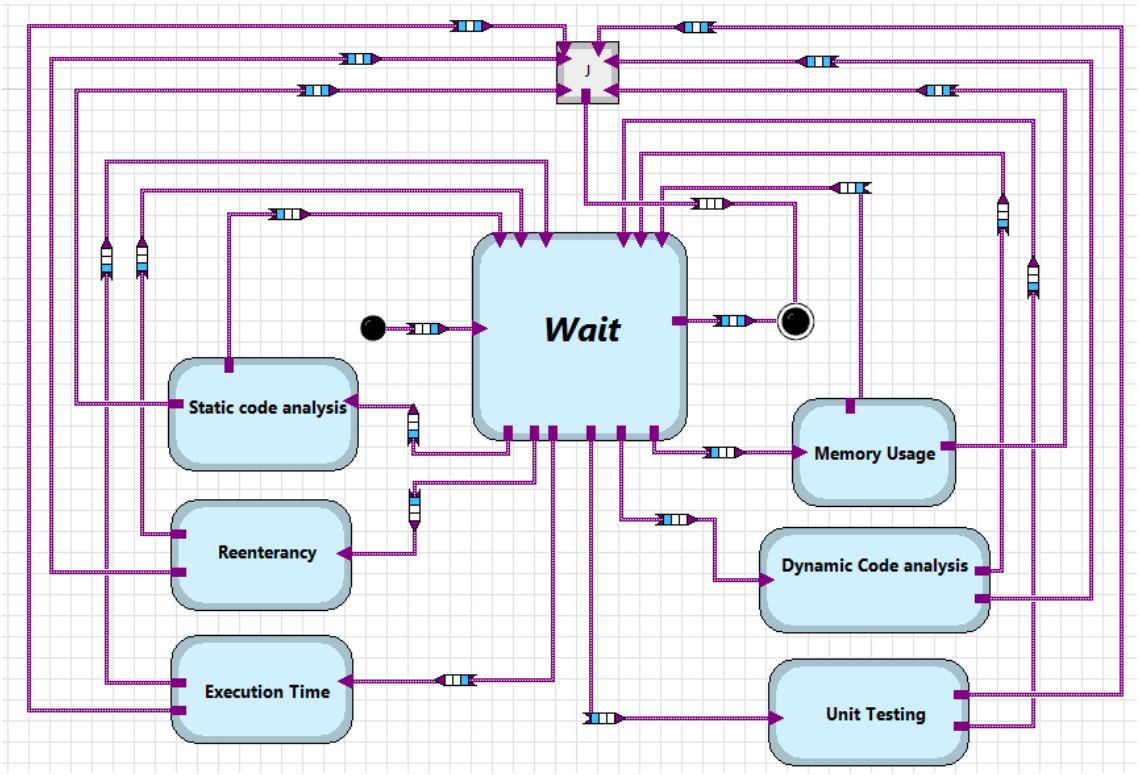


Abbildung A.15: Testaktivitäten in LabVIEW-Statechart

A.8. FPGA-Implementierung

Applikationsspezifischer Befehlssatz des LiSARD-Prozessors

Name	Ex. Command		WB. Command		Delay	Memory Read	Memory Write
ADD	Defaul	0	Add	3	4	1	1
SUB	Sub	1	Sub	4	4	1	1
MUL	Defaul	0	Mul	5	5	1	1
DIV	Defaul	0	Div	6	31	1	1
MOV	Defaul	0	Mov	2	1	1	1
IN	Defaul	0	In	0	0	0	1
OUT	Defaul	0	Out	1	1	1	0
DQ	None	2	None	7	0	0	0
mov_ISPOS	Fuse	3	G_IsPOS	16	1	1	1
mov_ISNEG	Fuse	3	G_IsNEG	17	1	1	1
mov_ISNULL	Fuse	3	G_IsNULL	18	1	1	1
Dbl2I16	Defaul	0	Dbl2Int	12	2	1	1
Clookup	Defaul	0	Mlookup	14	0	0	1

Tabelle A.4: Verwendete Befehle im Softcore-Prozessor

A.9. Messergebnisse

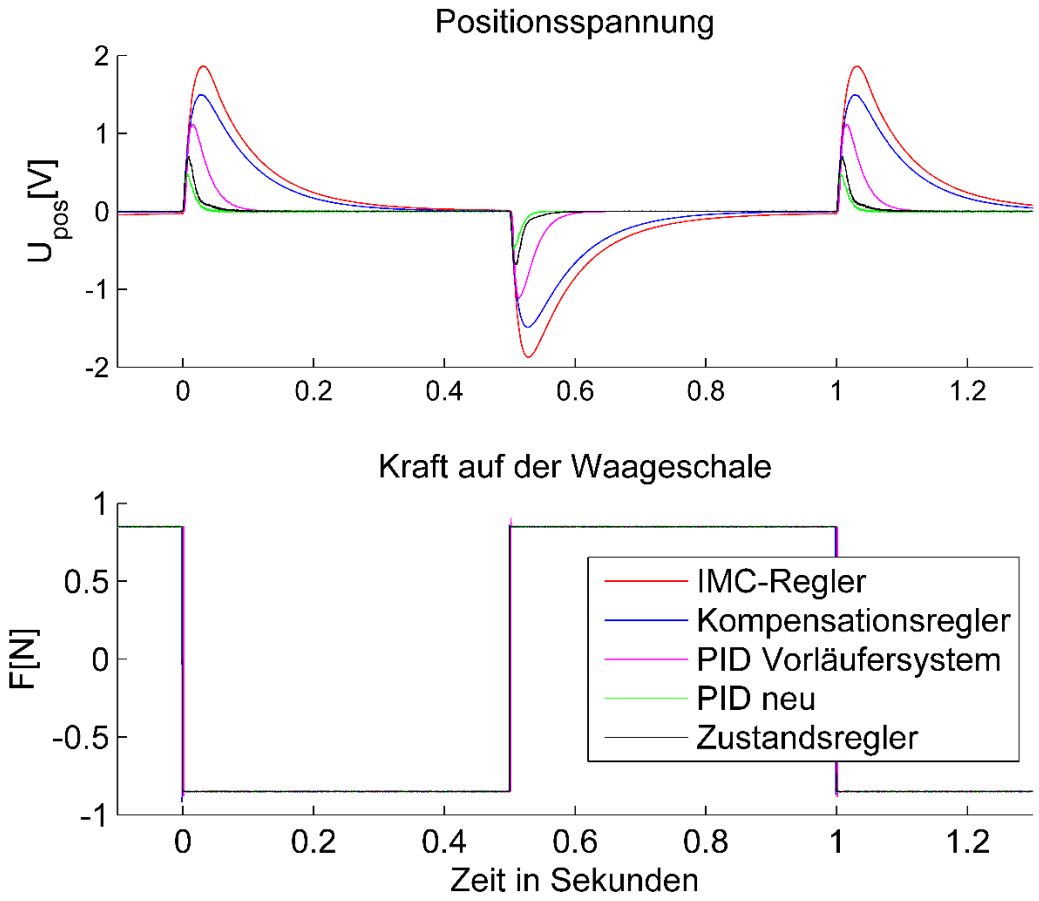


Abbildung A.16: Zeitverlauf der Positionsspannung während einer Lastwechselfolge mit 1 Hz Lastwechselfrequenz

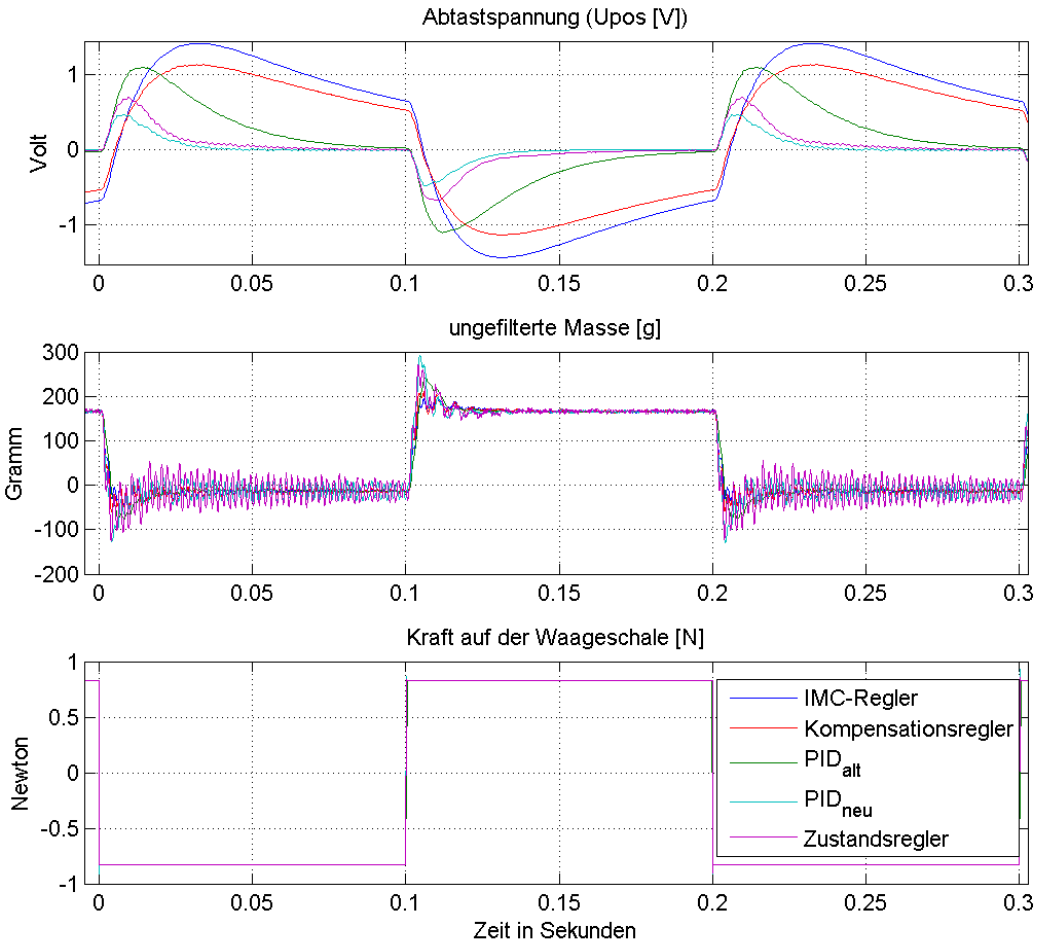


Abbildung A.17: Zeitverlauf der Messwerte während einer Lastwechselfolge (5Hz)

A.10. Ansätze zum Produktlinienentwurf

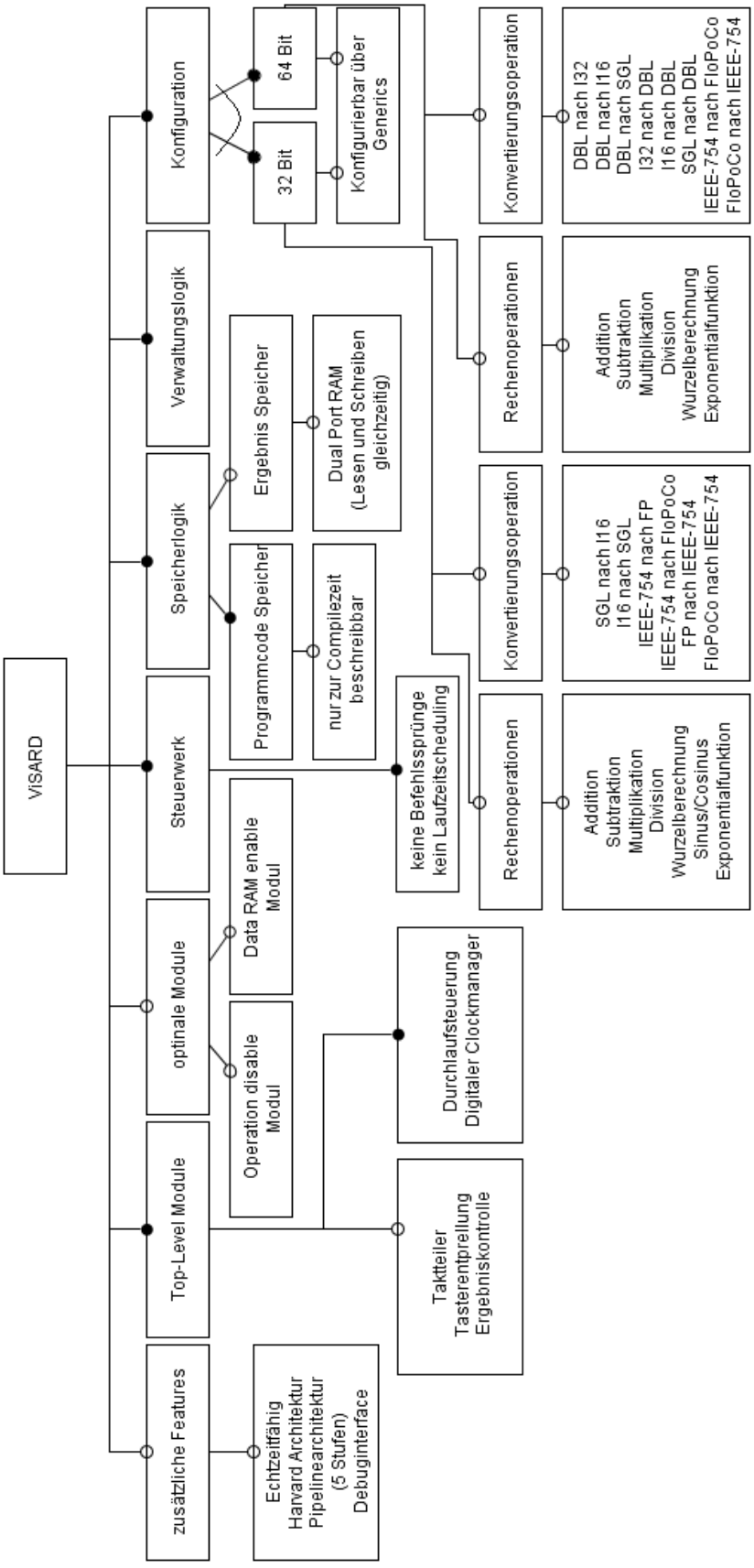


Abbildung A.18: ViSARD Feature Diagramm

Literaturverzeichnis

[Ag14] Springer Open Journal: „Fixed Point Theory and Applications“, Editor-in-Chief: Ravi P. Agarwal, ISSN: 1687-1812 (electronic version), Journal N 13663, 2014, http://www.fixedpointtheoryandapplications.com/content

[AKRW14] A. Amthor, I. Kaiser, N. Rogge, H. Weiß: "Analysis, physically motivated Modelling and system Identification of Electromagnetic Force Compensated (EMC)", 58th Ilmenau Scientific Colloquium, TU Ilmenau, 2014

[Alt15] Altera Corporation: DSP Builder Handbook, Volume 1: Introduction to DSP Builder, www.altera.com/literature/hb/dspb/hb_dspb_intro.pdf, 2015

[Am09] Amtsblatt der Europäischen Union: RICHTLINIE 2009/23/EG, L 122, 2009, http://www.lme.brandenburg.de/sixcms/media.php/4463/1_1_2_2_9_1_Waagenrichtlinie_2009.pdf, (Stand: 15.06.2014):

[Am10] A. Amthor: „Modellbasierte Regelung von Nanopositionier- und Nanomessmaschinen“, VDI Verlag, Düsseldorf 2010, ISSN 0178-9546, ISBN 978-3-18-517908-2

[Ar12] S. Arndt: „Semantische Optimierung im Requirements Engineering durch Terminologiemanagement“, in: *EKA2012 Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen, Magdeburg*, pp. 35-45, ISBN: 978-3-940961-72-3, 2012

[AUT15] AUTOSAR: AUTomotive Open System ARchitecture, http://www.autosar.org/

[B12] H. Bock: “Untersuchung der Realisierung von Verschlüsselungsalgorithmen mit FPGAs”, Diplomarbeit, TU Ilmenau, 2012

[B79] B. Boehm: „Guidelines for Verifying and Validating Software Requirements and Design Specifications“, Euro IFIP, P.A. Samet (editor), North-Holland Publishing Company, IFIP,1979, s. 711-719

[B95] E. Best: „Semantik. Theorie sequentieller und paralleler Programmierung“, Lehrbuch Informatik, Vieweg, Braunschweig/Wiesbaden, 1995, ISBN 13: 9783528054311

[B98] H. Balzert: „Lehrbuch der Software-Technik. Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung“, Spektrum Akademischer Verlag, Berlin 1998, ISBN 3-8274-0065-1

[Ba15] H. S. Baumgartl: „Optimierung dynamischer Waagen nach dem Prinzip der elektromagnetischen Kraftkompensation mittels numerischer Modelle zur Systemsimulation“, Dissertation, TU Ilmenau, 2015

[Bal14] V. Balasubramanyam: “Model-based development and testability for information processing on FPGA in PXI-system”, Masterarbeit, TU Ilmenau, 2014

[Bau09] Baumann, Tommy: „Beiträge zur Automatisierung der frühen Entwurfsphasen verteilter Systeme“, Dissertation, TU Ilmenau, 2009, http://www.db-thueringen.de/servlets/DerivateServlet/Derivate-18245/ilm1-2009000122.pdf

[BCV06] K. Bertels, J.M.P. Cardoso, S. Vassiliadis: “Reconfigurable Computing: Architectures and Applications”, Second International Workshop ARC 2006, Springer-Verlag Berlin Heidelberg, ISSN 0302-9743, 2006

[Be05] K. Bender: „Embedded Systems-qualitätsorientierte Entwicklung“, Springer-Verlag Berlin Heidelberg 2005, ISBN 3-540-22995-7

[Be91] R. Best: „Digitale Meßwertverarbeitung“, München, Wien: Oldenbourg 1991, ISBN 3-486-21573-6

[Be96] Ch. Berg: „Grundlagen der Wägetechnik – Begriffe, Meßverfahren, Fehlereinflüsse“, Sartorius AG, 2. Auflage 1996

[BHH95] W. Balachandran, M. Halimic, M. Hodzic, M. Tariq, Y. Enab, F. Cecelja: “Optimal digital Control and Filtering for Dynamic Weighing Systems”, Proceedings of Instrumentation and Measurement Technology Conference IMTC, IEEE, 1995

[Bie07] C. Bieser: „Konzept einer bibliotheksbasiert konfigurierbaren Hardware-Testeinrichtung für eingebettete elektronische Systeme“, Dissertation, Universität Fridericiana Karlsruhe 2007, ISBN 978-3-937295-78-7

[BK08] C. Bunse, A. von Knethen: „ Vorgehensmodelle kompakt“, Heidelberg 2008, ISBN 978-3-8274-1950-7

[BKPS04] G. Böckle, P. Knauber, K. Pohl, K. Schmid: "Software-Produktlinien: Methoden, Einführung und Praxis", 2004, ISBN 3-89864-257-7

[Bo88] K.W. Bonfig: „Technische Druck- und Kraftmessung“, expert Verlag 1988, ISBN 3-8169-0315-0

[Bo99] B.W. Bomar: “Finite Wordlength Effects”, in *V.K. Madisetti und D.B. Williams: “Digital Signal Processing Handbook”*, CRC Press LLC, 1999

[BR13] B. Bundschuh, I. Rennert: „Signale und Systeme: Einführung in die Systemtheorie“, Carl Hanser Verlag GmbH & Co. KG, 2013, ISBN-10: 3446433279, ISBN-13: 978-3446433274

[Br96] R. Brigola: “Fourieranalysis, Distributionen und Anwendungen: Ein Einstieg für Ingenieure, Naturwissenschaftler und Mathematiker”, Vieweg Verlagsgesellschaft 1996, ISBN-10: 3528066199, ISBN-13: 978-3528066192

[BRS13] D. F. Bacon, R. Rabbah, S. Shukla: „FPGA Programming for the Masses“, 2013 ACM 1542-7730/13/0200, doi 10.1145/2436696.2443836

[BS10] H. Bäumel, R. Schrod: “Eichfähiges Wägesystem und Verfahren zur Ermittlung eichpflichtiger Messwertdaten”, Patent DE 10041251B4 11.03.2010

[BS91] W.-J. Becker, P. Siebert: „Elektromechanische Kompensationswaage mit digitaler Regelung“, wägen+dosieren 6/1991

[BST10] K. Berns, B. Schürmann, M. Trapp: „Eingebettete Systeme: Systemgrundlagen und Entwicklung eingebetteter Software“, Vieweg+Teubner Verlag 2010, ISBN 978-3-8348-0422-8

[CG13] C. Corradi, Xilinx, R. Girardey: “Developing FPGA applications for Edition 2 of the IEC 61508 Safety Standard”, 2013, http://www.eetimes.com/document.asp?doc_id=1280367

[Ch14] A. Chandrasekaran: „Special Development System for Information Processing in Measurement Domain“, Research Project, TU Ilmenau, 2014

[CK13] M. Chupilko, A. Kamkin: “Runtime Verifikation Based on Executable Models: On-the Fly Matching of Timed Traces”, in *Model Based Testing (MBT13), EPTCS 111, 2013*, pp. 67-81, doi:10.4204/EPTCS.111.6. http://rvg.web.cse.unsw.edu.au/eptcs/Published/MBT2013/Papers/4/arXiv.pdf

[Co04] M. Conrad: „Modell-basierter Test eingebetteter Software im Automobil: Auswahl und Beschreibung von Testszenarien“, Dissertation, TU Berlin, 2004, ISBN 3-8244-2188-7

[CTB14] CTB Embedded Systems Guide: „DO-178B Certification Standard for Avionics“, DO-254 Design Assurance Guidance for Airborne Electronic Hardware“,http://www.embedded-systems-portal.com/CTB/DO-178B_Certification_Standard_for_Avionics,10035.html (Stand: 8.08.2014)

[DH01] V. Duridanova, T. Hummel: “Modelling of Embedded Mechatronic Systems Using Hybrid Petri Nets”, in *M. H. Hamza (Ed.): “IASTED International Conference Modelling, Identification, and Control”*, 2001, Austria, IASTED/ACTA Press 2001, vol. 1, pp. 521-526. ISBN 0-8898-6316-4

[DIN1319] Die grundlegende Deutsche Norm der Messtechnik, DIN 1319-1: „Grundlagen der Messtechnik-Grundbegriffe“, DIN 1319-2: „Grundlagen der Messtechnik-Begriffe für Messmittel“, http://www.eu-richtlinien-online.de/cn/bGV2ZWw9dHBsLWVudGhhbHRlbmVkb2NzJmFyd-GlkPTE1MzgyODA3Mg**.html, (Stand 4.11.2013)

[DLJ99] B. P. Dave, G. Lakshminarayana, N. K. Jha: ”COSYN: Hardware-software cosynthesis of heterogeneous distributed embedded systems”. *IEEE Trans., VLSI Systems*, 7(1):92–104, Mar. 1999

[DN06] DIN Deutsches Institut für Normierung: „DIN EN 45020: Normierung und damit zusammenhängende Tätigkeiten- Allgemeine Begriffe (identisch mit: ISO/IEC Guide 2)“, Berlin, 2006

[DN92] Deutsche Norm: „Metrologische Aspekte nichtselbsttändiger Waagen, DIN EN 45501“, 1992, http://legnet.metas.ch/legnet2/Eichaemter/Allgemeine_Informationen/Normen%20%28Hidden-kein%20Zugriff%29/DIN_EN_45501.pdf (Stand: 3.11.2014):

[Do08] S. Dontsova: “Digitale Signalverarbeitung in der dynamischen Wägetechnik”, Dissertation, Ilmenau, ISLE-Verlag, 2008

[DPF12] B. Däne, A. Pacholik, W. Fengler: “Aspekte zur Realisierung eines FPGA-basierten Softcore-Prozessors für den Einsatz in Regelungs- und Messsystemen”, in *EKA 2012 "Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen"*, ISBN: 978-3-940961-72-3, 2012, Magdeburg

[DPZF13] B. Däne, A. Pacholik, S. Zschäck, W. Fengler, C. Ament, T. Braune: „Designing a control applicationby using a cpecialized multi-core soft microprocessor“, in *Z.Slanina, 12th IFAC Conference on Pragrammable Devices and Embedded Systems (PDeS 2013)*, pp.221-226, Velke Karlovice, Czech Reoublic, 2013

[DR14] J. Daemen, V. Rijmen: „AES Proposal: Rijndael“, (Stand 5.12.2014) http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf

[EGK08] C. Eck, H. Garcke, P. Knabner : „Mathematische Modellierung“, Springer, Berlin 2008; Auflage: 1, ISBN-10: 3540749675, ISBN-13: 978-3540749677

[ESB07] J. Ensthaler, K. Strübbe, L. Bock: „Zertifizierung und Akkreditierung technischer Produkte: ein Handlungsleitfaden für Unternehmen“, Springer Verlag, 2007, ISBN-10: 3540694358, ISBN-13: 978-3540694359

[Fa10] M. Farrag: “Colored Model Based Testing for Software Product Lines (CMBT-SWPL)”. PhD thesis, TU Ilmenau, 2010

[FDA14] U.S FDA Home-page: http://www.fda.gov, (Stand 20.12.2014)

[FFH02] E. Fuchs, K.H. Fuchs, C.H. Hauri: „Requirements-Engineering in IT effizient und verständlich“, Vieweg Verlag Wiesbaden, 1. Auflage, 2002

[Fin94] H. Finger: „Elektrische Wägetechnik, VEB Verl. Technik Berlin, 1964

[FP91] W. Fengler, I. Philippow: „Entwurf industrieller Microcomputersysteme“, München, Wien: Hanser, 1991, ISBN 3-446-16150-3

[FPL13] ENS Lyon: FPLibrary, http://babbage.cs.qc.cuny.edu/IEEE-754/, Version 2013

[FPM06] W. Fengler, A. Pacholik, M. Mologina: “Modeling Technique for Model Based Error Localization and Error Removal Based on Extended UML Activity Diagrams”, in: *M. Adamski, L. Gomes, M. Węgrzyn, G. Łabiak (Eds.): DESDes'06 - Discrete Event System Design*, ISBN 83-7481-035-1, IFAC WS 2006 0011 PL, University of Zielona Góra Press, 2006

[G08] I. Grout: „Digital Systems Design with FPGAs and CPLDs“, Elsevier Ltd, 2008, ISBN-13: 978-0-7506-8397-5

[GCh01] K. Gaj, P. Chodowiec: “Fast implementation and fair comparison of the final candidates for Advanced Encryption standard using field Programmable Gate Arrays”, RSA Security Conference – Cryptographer´s Track, San Francisco, 2001

[GDMF12] I. Gushchina, B. Däne, A. Moskalev, W. Fengler: „Untersuchung zur FPGA-Implementierung von Mess- und Regelungsalgorithmen“, in: *EKA2012 Beschreibungsmittel, Methoden, Werkzeuge und Anwendungen, Magdeburg*, pp. 111-119, ISBN: 978-3-940961-72-3, 2012

[GM07] R. Gessler, Th. Mahr: „Hardware-Software-Codesign: Entwicklung flexibler Mikroprozessor-FPGA-Hochleistungssysteme“, Wiesbaden 2007, ISBN 978-3-8348-0048-0

[Go14] I. Goncharova: “Untersuchungen zum H∞-Reglerentwurf für eine Präzisionswaage“, Masterarbeit, TU Ilmenau, 2014

[GT02] P. Gerrard, N. Thompson: „Risk-Based E-Business Testing“, Artech House INC 2002, ISBN-13: 9781580533140, ISBN-10: 1580533140

[H87] D. Harel: „ Statecharts: A visual formalism for complex systems“, Science of Computer Programming, 1987

[He08] S. Helke: „Verifikation von Statecharts durch struktur- und eigenschaftserhaltende Datenabstraktion“, Dissertation TU Berlin, 2008, https://www.deutsche-digitale-bibliothek.de/binary/RPEH5WDYQPLH2GIWLPLTY3YDSLS5BQJD/full/1.pdf

[HH08] R. Höhn, S. Höppner: „Das V-Modell XT: Anwendungen, Werkzeuge, Standards“, Springer-Verlag Berlin Heidelberg 2008, ISBN 978-3-540-30249-0, DOI 10.1007/978-3-540-30250-6

[HHR12] P. Harpe, H. Hegt, A. van Roermund: „Smart AD and DA Conversion“, Springer 2012, ISBN-10: 9400732570, ISBN-13: 978-9400732575

[HSD09] H. Höhn, B. Sechser, K. Dussa-Zieger, R. Messnarz, B. Hindel: “Software Engineering nach Automotive SPICE“, Heidelberg 2009, ISBN 978-3-89864-578-2

[Hu12] T. V. Huynh: “Efficient Floating-Point Implementation of Signal Processing Algorithms on Reconfigurable Hardware”, Dissertation, Graz University of Technology, Austria, 2012

[Hü96] Hütte: die Grundlagen der Ingenieurwissenschaftler, Akademischer Verein Hütte e.V., Berlin 1996, Hrsg. von Horst Czichos. – 30. neubearb. und erw. Aufl.-Berlin, ISBN 3-540-58740-3

[HV-H02] P. Holleczek, B. Vogel-Heuser: PEARL 2002: „Sicherheit und Verfügbarkeit in Echtzeit- und Automatisierungssystemen“, Springer-Verlag 2002, ISSN 1431-472-X, ISBN 3-540-44332-0

[IE14] Institute of Electrical and Electronics Engineers, Inc. (2008): “IEEE Standard for Floating-Point Arithmetic", (Stand: 10.10.2014) http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4610935

[Ja10] K. Janschek: „Systementwurf mechatronischer Systeme“, Springer-Verlag Berlin Heidelberg 2010, ISBN 978-3-540-78876-8, e-ISBN 978-3-540-78877-5, DOI 10.1007/978-3-540-78877-5

[Jä94] G. Jäger: „Ein Beitrag zur dynamischen Wägetechnik“, wägen+dosieren 2/1994

[JBK09] K. Jensen, J. Billington, M. Koutny (Eds.): “Transactions on Petri Nets and Other Models of Concurrency III”, Springer Berlin Heidelberg NewYork 2009, ISBN-10 3-642-04854-4, ISBN-13 978-3-642-04854-8

[JCGM10] JCGM 100:2008, GUM 1995 with minor corrections: „Evaluation of measurement data - Guide to the expression of uncertainty in measurement“, Corrected version 2010,
http://www.bipm.org/utils/common/documents/jcgm/JCGM_100_2008_E.pdf

[JH12] R. Jamal, R. Heinze: “Virtuelle Instrumente in der Praxis: Mess-, Steuer-, Regel- und Embedded-Systeme 2012”, Begleitband zum 17.VIP-Kongress, VDE Verlag GmbH, ISBN 978-3-8007-3412-2, 2012

[K08] A. Korff: “Modellierung von eingebetteten Systemen mit UML und SysML”, Spektrum akademischer Verlag Heidelberg 2008, ISBN 978-3-8274-1690-2

[K11] G. Kemnitz: „Technische Informatik - Band 2:Entwurf digitaler Schaltungen“, Springer-Verlag Berlin Heidelberg 2011, ISSN 1614-5216, ISBN 978-3-642-17447-6, DOI 10.1007/978-3-642-17447-6

[KCF04] T. Klein, M. Conrad, I. Fey, M. Grochtmann: „Modellbasierte Entwicklung eingebetteter Fahrzeugsoftware bei DaimlerChrysler“, Lecture Notes in Informatics (LNI), 2004

[Kir13] M. Kirchhoff: „Entwicklung von ASIC-Lösungen auf Basis von FPGA-Prototypen“, Hauptseminar, TU Ilmenau, 2013

[Kir14] M. Kirchhoff: „Methodik und Spezialisierung von Softcore-Prozessoren auf Basis von VHDL“, Masterarbeit, TU Ilmenau, 2014

[KKF08] J. Klöckner, S. Köhler, W. Fengler: “Model Based Design of Networked Embedded Systems”, in: *ICINCO 2008. 5th International Conference on Informatics in Control, Automation and Robotics.* Funchal, Madeira - Portugal, INSTICC Press, 2008, S. 253–259

[Kl09] S. Kleuker: “Formale Modelle der Softwareentwicklung: Model-Checking, Verifikation, Analyse und Simulation”, Vieweg+Teubner Verlag, 2009, ISBN-10: 3834806692, ISBN-13: 978-3834806697

[Ko13] M. Kozlachkova: „Entwicklung von Verifikationsmethoden auf Basis von Petri-Netzen für die Fehler-, Zeit- und Ressourcenanalyse in der komplexen Informationsverarbeitungssystemen“, Masterarbeit, TU Ilmenau, 2013

[Ko89] M. Kochsiek: „Handbuch des Wägens“, 2. Bearbeitete und erweiterte Auflage, Friedrich Vieweg & Sohn Verlagsgesellschaft GmbH Braunschweig/Wiesbaden, 1989

[Kr04] L. Krause: „Dynamische Wägetechnik“, Ilmenau, Habilitationsschrift, Wissenschaftlicher Verlag Berlin 2004, ISBN-10: 3865730612, ISBN-13: 978-3865730619

[Kra14] N. Krauße: „Angepasster Codegenerator für ViSARD“, Hauptseminar, 2014

[Li04] T. Licht: „Ein Verfahren zur zeitlichen Analyse von UML-Modellen beim Entwurf von Automatisierungssystemen“, Dissertation, TU Ilmenau, ISBN 3-932633-91-1, 2004

[LL10] J. Ludewig, H. Lichter: „Software Engineering: Grundlagen, Menschen, Prozesse, Techniken“, Heidelberg, 2010, ISBN 978-3-89864-662-8

[LPJW08] S. Lämmermann, A. Pacholik, A. Jesser, R. Weiss, J. Ruf, W. Fengler, L. Hedrich, T. Kropf, W. Rosenstiel: „Improving Mixed-Signal Verification by Assertion Based Design“, in *16th IFIP/IEEE International Conference on Very Large Scale Integration VLSI-SoC 2008*, Rhodos, Griechenland

[LR05] P. Liggesmeyer, D. Rombach: „Software Engineering eingebetteter Systeme – Grundlagen, Methodik, Anwendungen“, München 2005, ISBN 3-8274-1533-0

[LSR07] F. Van Der Linden, K. Schmid, E. Rommes: “Software product lines in action: the best industrial practice in product line engineering”, Springer-Verlag New York, Inc, 2007

[Lü09] T. Lübbert: „Integrierte Entwurfs-und Verifikationsmethode für verteilte eingebettete Steuerungssysteme am Beispiel eines Flugzeugkabinensystems“, Dissertation, Universität Bochum, 2009, http://www-brs.ub.ruhr-uni-bochum.de/netahtml/HSS/Diss/LuebbertTim/diss.pdf

[LV15] Systemdesign Software NI LabVIEW http://www.ni.com/labview/d/, Internet Seite von National Instruments, (Stand 17.2.2015)

[M13] Internetseite MathWorks: http://www.mathworks.de/, (Stand 17.2.2015)

[Mä09] S. Mäuselein: “Untersuchungen an Silizium-Verformungskörpern für die Anwendung in der Präzisions-Kraftmess- und Wägetechnik“, Dissertation, TU Ilmenau, 2009

[Ma11] P. Marwedel: „Embedded Systems Design: Embedded Systems Foundations of Cyber-Physical Systems“, Springer Science+Business Media 2011, ISBN 978-94-007-0256-1, e-ISBN 978-94-007-0257-8, DOI 10.1007/978-94-007-0257-8

[Mae09] P. Mäder: „Rule-Based Maintenance of Post-Requirements Traceability“, TU Ilmenau, Dissertation, 2009

[MAFA09] M. Müller, A. Amthor, W. Fengler, C. Ament: “Model-driven Development and Multiprocessor Implementation of a Dynamic Control Algorithm for Nanopositioning and Nanomeasuring Machines”, in: *Journal of System and Control Engineering (JSCE), Professional Engineering Publishing, London/UK* Bd. 223, Professional Engineering Publishing, London/UK, 2009

[Mam14] R. Mambally Das: “Model-Based Analysis of Measurement Uncertainty for ADCs”, Research project, TU Ilmenau, 2014

[MB07] U. Meyer-Baese: „Digital Signal Processing with Field Programmable Gate Arrays“, Springer-Verlag Berlin Heidelberg 2007, ISBN 978-3-540-72612-8

[ME09] Arbeitsgemeinschaft Mess- und Eichwesen (AG ME): Richtlinie 2004/22/EG für Messgeräte (MID) Informationen für Verwender von Messgeräten, 2009: http://www.lbme.nrw.de/download/AGME_Infobl_MID_09.pdf, (Stand: 5.09.2014)

[MJZ13] P. Mäder, P.L. Jones, Y. Zhang, J. Cleland-Huang: „Strategic Traceability for Safety-Critical Projects“, IEEE Software, IEEE Computer Society 2013

[MKGP13] M. Müller, J. Klöckner, I. Gushchina, A. Pacholik, W. Fengler, A. Amthor: "Performance evaluation of platform-specific implementations of numerically complex control designs for nano-positioning applications", *in: International journal of embedded systems : IJES. - Genève : Inderscience Publ.. - Vol. 5.2013, 1/2, pp. 95-105*

[MP14] C.-F. Müller: „Zertifizierung eines FPGA-basierten Systems: Anforderungen an das Informationsverarbeitungssystem“, Projektseminar, TU Ilmenau, 2014

[MS89] R. Maier, G. Schmidt: “Advanced Digital Control and Filtering for High-Precision Weighing Cells”, Proceedings of the International Conference on Advanced Mechatronics, 1989

[MSch89] R. Maier, G. Schmidt: „Integrated Digital Control and Filtering for an Electromagnetically Compensated Weighing Cell“, IEEE Transactions on Instrumentation and Measurement, Vol. 38, No. 5., 1989

[Mü12] M. Müller: „Beitrag zum modellbasierten Entwurf eingebetteter Systeme: Komponentenbasierte Modellierungsansätze für verteilte rekonfigurierbare Plattformen“, Dissertation, TU Ilmenau, 2012

[MW15] MathWorks: „Using FDATool“, (Stand: 8.01.2015): http://de.mathworks.com/help/signal/ug/opening-fdatool.html

[Nag14] S. Nagaraj: „Design and Implementation of a special test environment for LiSARD softcore programs“, Research Project, TU Ilmenau, 2014

[Neu04] H. W. Neukirchen: „Languages, Tools and Patterns for the Specification of Distributed Real-Time Tests“, Göttingen, Georg-August-Universität, 2004

[Ng10] H. Nguyen: „Optimierung verschiedener digitaler Waagen für dynamische EMK-Waagen“, Diplomarbeit, TU Ilmenau, 2010

[NI14] National Instruments, Online-Katalog: „Wie funktioniert FPGA“, www.ni.com (Stand 6.3.2014)

[NI15] NI LabVIEW Statechart Module, http://www.ni.com/labview/d/statechart, 2015

[NM10] G. Nicolescu, P.J. Mosterman: „Model-Based Design for Embedded Systems“, Taylor&Francis Group, Boca Raton USA, 2012, ISBN 978-1-4200-6784-2

[O14] OIML International Organization of Legal Metrology, Home Page: https://www.oiml.org, (Stand: 8.10.2014)

[OC15] OpenCores: development of hardware IP cores as open source, http://opencores.org

[Öl15] D. Ölschlegel: “Erstellung einer design-Bibliothek für grafische Funktionen mit Code-generator für eine Assemblersprache”, TU Ilmenau, 2015

[P13] C. Pohl: Modellierung und Simulation von externen Erweiterungen und Schnittstellen eines FPGAs und deren Integration in ein Applikationsmodell“, Bachelorarbeit, 2013

[P75] C. A .Petri: Interpretations of a net theory“, Technical report 75-07, GMD, Bonn, 1975

[Pa09] A. Pacholik: „Entwicklung und Gegenüberstellung von Methoden zur automatisierten Verifikation von ausführbaren Systemspezifikationen“, Dissertation, TU Ilmenau, 2009

[Pau14] P.K. Paulsami: „Product line for embedded systems in the measurement domain“, Research Project, TU Ilmenau, 2014:

[PBL05] K. Pohl, G. Böckle, F. J. van der Linden: „Software Product Line Engineering: Foundations, Principles and Techniques”, Springer Verlag Berlin Heidelberg, 1 edition, 2005. ISBN 978-3642063640

[PC15] PENECA chromos, http://tin.tu-ilmenau.de/ra/skripte/pn-_chr/,1997

[Pf96] A. Pfeiffer: „Integrierter H_∞-Regler/Filterentwurf für elektromechanische Präzisionswaagen“, Dissertationsschrift, Düsseldorf, VDI-Verlag, 1996

[PKMG11] A. Pacholik, J. Klöckner, M. Müller, I. Gushchina, W. Fengler: "LiSARD: LabVIEW Integrated Softcore Architecture for Reconfigurable Devices", in: *International Conference on Reconfigurable Computing and FPGAs (ReConFig '11)*, Nov. 30 - Dec. 2, 2011, Cancun, Mexico: IEEE Computer Society CPS, pp. 442-447

[PP94] P. Profos, T. Pfeifer: „Handbuch der industriellen Meßtechnik“, 6. Durchgesehene und korrigierte Auflage, R.Oldenbourg Verlag München Wien 1994, ISBN 3-486-22592-8

[PS94] A. Pfeiffer, G. Schmidt: „Integrated H^∞ Design for Filtering and Control Operations of a High-Precision Weighing Cell“, Proceedings of the Asian Control Conference, Vol. 2, 1994

[PTB06] PTB-Arbeitsgruppe 1.14 "IT-Wägetechnik: Physikalisch-Technische Bundesanstalt Merkblatt über die Anforderungen an die Softwaredokumentation“, 2006

[PTB14] PTB Home-page: „Waagen, Arbeitsgruppe 1.12“: http://www.ptb.de/cms/fachabteilungen/abt1/fb-11/ag-112.html

[R07] C. Rupp, SOPHIST Group: „Requirements-Engineering und –Management– Professionelle, iterative Anforderungsanalyse für die Praxis“, 4.Auflage, Carl Hanser Verlag München Wien, Nürnberg, 2007

[Ra13] K. Raman: “Data and Design Integrity for Security of Realization on FPGA (PXI-System)”, Masterarbeit, TU Ilmenau, 2013

[Ram14] Ramyashree: „Model based analysis of errors in signal processing“, Research Project, TU Ilmenau, 2014

[RBGW10] T. Roßner, C. Brandes, H. Götz, M. Winter: „Basiswissen modellbasierter Test“, dpunkt Verlag, Heidelberg, 2010, ISBN 3-898-64589-4

[RDK06] M. Russ, B. Danzer, D. Korotkiy: „Virtueller Funktionstest für die Entwicklung von eingebetteter Software“, A&D Kompendium 2005/2006, pp.75-76

[Re10] W. Reisig: „Petrinetze: Modellierungstechnik, Analysemethoden“, Fallstudien, Leitfäden der Informatik. Vieweg+Teubner, 2010; ISBN 978-3-8348-1290-2

[RIO15] National Instruments: “Multifunktions-RIO-Hardware der R-Serie von National Instruments”, http://sine.ni.com/nips/cds/view/p/lang/de/nid/11829

[RN05] A. Rausch, D. Niebuhr: „Erfolgreiche IT-Projekte mit dem V-Modell XT“, OBJEKTspektrum 3/2005 (Mai 2005), http://agrausch.informatik.uni-kl.de/publikationen/repository/journal/jour015/niebuhr_rausch_OS_03_05.pdf

[Ro91] S. Roocke: „Ein Beitrag zur Erhöhung der Genauigkeit von interferenzoptischen Kraftmeßzellen“, Dissertation, TU Ilmenau, 1991

[Ro99] P. Rokyta: „ESDA-basierter System- und ASIC-Entwurf mit Petri-Netzen“, Dissertation, TU Ilmenau, 1999

[Roz13] E. Rozova: „Modifikation und Weiterentwicklung des Softcore-Prozessors LiSARD“, Masterarbeit, TU Ilmenau, 2013

[RSG01] C. Rupp, Sophist Group: “Requirements-Engineering und –Management: Professionelle, iterative Anforderungsanalyse für IT-Systeme”, Hanser Verlag, München, 2001, ISBN 3-446-21664-2

[RZ04] M. Reuter, S. Zacher: „Regelungstechnik für Ingenieure: Analyse, Simulation und Entwurf von Regelkreisen“, Vieweg+Teubner Verlag, 2004, ISBN-10: 3528050047, ISBN-13: 978-3528050047

[Sa12] Samson AG - Mess- und Regeltechnik: „Technische Information-Digitale Signale“, Teil 1 –L150: Grundlagen, http://www.samson.de/pdf_de/l150de.pdf

[Sa13] M. Sauer: „Entwurf und Realisierung eines Hardware-Testkonzeptes für einen Softcore-Prozessor (Realisierung in VHDL)“, Bachelorarbeit, TU Ilmenau, 2013

[Sch12] R. Schulze: “Formale Verifikation eines 8051-Mocrocontrollers”, Masterarbeit, TU Ilmenau, 2012

[Sch13] M. Schaible: „Towards the Verification of a Table-driven Microprocessor Architecture for Safety-critical Systems”, in *Unger H., Halang W.A. “Autonomous Systems 2013”*, VDI Verlag GmbH, Düsseldorf, 2013, ISBN 978-3-18-382710-7

[SFM12] H. Salzwedel, N. Fischer, S. Marwedel: “Model Based Development of Networked Electronics in Aircraft and Automobiles”, *DAC Workshop on System Level Design of Automotive Engineering Systems, SLDAES2012*, San Francisco, California, 2012

[Si04] D.E. Simanek: „Propagation of Errors“, Online Version (November 2014): https://www.lhup.edu/~dsimanek/scenario/errorman/propagat.htm

[SKWW98] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, N. Ferguson: “Twofish: A 128-Bit Block Cipher”, https://www.schneier.com/paper-twofish-paper.pdf, 1998

[SL12] A. Spillner, T. Linz: „ Basiswissen Softwaretest“, dpunkt Verlag, Heidelberg, 2012, ISBN: 978-3-86490-024-2

[SRWL11] A. Spillner, T. Roßner, M. Winter, T. Linz: “Praxiswissen Softwaretest-Testmanagement”, Heidelberg 2011, ISBN 978-3-89864-746-5

[St04] D. Streitferdt. Family-Oriented Requirements Engineering. PhD thesis, Technical University of Ilmenau, 2004, http://www.db-thueringen.de/servlets/DerivateServlet/Derivate-2763/ilm1-2004000032.pdf

[SZ03] J. Schäuffele, T. Zurawka: „Automotive Software Engineering“, 2003, Vieweg, Wiesbaden, ISBN 3-528-01040-1

[T15] TIme petri Net Analyzer TINA, http://projects.laas.fr/tina//home.php

[Tei97] J. Teich: „Digitale Hardware/Software Systeme: Synthese und Optimierung“, Springer-Verlag Berlin Heidelberg 1997, ISBN 3-540-62433-3

[TNRW01] G. Thiele, R. Neimier, I. Renner, E. Wendland, G. Schulz-Ekloff: „Zur Entwicklung zertifizierbarer fehlertoleranter Realzeit-Software mit Funktionsblock-Diagrammen für die Automatisierung“, in P.Holleczek, B.Vogel-Heuser (Hrsg.): *Proc. PEARL 2001, Echtzeit-Kommunikation und Ethernet/Internet*, Springer-Verlag Berlin Heidelberg 2001

[Tr15] Revere Transducers: Eichvorschriften, in: Anwender-Info 09/4-10D/01, http://www.zelo.biz/downloads/pdf/zelo/Eichvorschriften.pdf (Stand 16.2.2015)

[UL07] M. Utting, B. Legeard: „Practical model-based Testing: a Tools Approach“, 1st Edition, Amsterdam: Elsevier/Morgan Kaufmann, 2007, ISBN-10: 0123725011, ISBN-13: 978-0123725011

[VA14] Mentor Graphics, Verification Academy: “Assertion-Based Verification”, Online courses: https://verificationacademy.com/courses/assertion-based-verification (Stand: 3.09.2014)

[Val14] G.J. Vallant: “Modellbasierte Entzerrung von Analog/Digital-Wandler-Systemen”, Dissertation, Karlsruher Institut für Technologie, 2014, ISSN: 1433-3821

[W12] K. Wenzel: „Testkonzeption, Modellbasierte Validierung, Hardware-Co-Simulation und Integration von FPGA-Komponenten für Weißlichtinterferometrieanwendung“, Masterarbeit, TU Ilmenau 2012

[Wei12] M. Weichenhain: „Requirements-Engineering und Projektmanagement in der dynamischen Wägetechnik“, Masterarbeit, TU Ilmenau, 2012

[WEL04] European cooperation in legal metrology WELMEC: "Software Guide (Measuring Instruments Directive 2004/22/EC)", WELMEC 7.2, http://www.welmec.org/fileadmin/user_files/publications/7-2_Issue4.pdf 2004

[WEL12] WELMEC: „WELMEC 8.8: Leitfaden zu den allgemeinen und verwaltungstechnischen Aspekten des freiwilligen Systems zur modularen Bewertung von Messgeräten“, Ausgabe 2, Ljubljana/Slowenien, 2012

[WEL14] European Cooperation in Legal Metrology WELMEC, Home page http://www.welmec.org/ (Stand: 5.09.2014)

[WGA13] H. Weis, I. Gushchina, A. Amthor; F. Hilbrunner, T. Fröhlich: “Investigation of digital control concepts for dynamic applications of electromagnetic force compensated balances”, in: *2013 NCSLI International Workshop and Symposium*, 2013

[WW10] N. Weichert, M. Wülker: „Messetechnik und Messdatenerfassung“, Oldenbourg Wissenschaftsverlag GmbH, München2010, ISBN 978-3-486-59773-8

[Xil11] Xilinx, Inc.: “Xilinx System Generator for DSP User Guide”, 2011

[Xil14] Xilinx, Inc.: “Xilinx CORE Generator System”, http://www.xilinx.com/tools/coregen.htm (Stand: 5.06.2014)

[Zi08] A. Zimmermann: „Stochastic Discrete Event Systems: Modeling, Evaluation, Applications“, Springer Verlag, 2008, ISBN 978-3-540-74173-2

[ZKGA13] S. Zschäck, J. Klöckner, I. Gushchina, A. Amthor, C. Ament: “Control of nanopositioning and nanomeasuring machines with a modular FPGA based data processing system”, in *IFAC Mechatronics*, 23(3): pp. 257-263, 2013

[Zu06] R. Zurawski: „Embedded System handbook“,Taylor&Francis Group, 2006, ISBN-10: 0-8493-2824-1, ISBN-13: 978-0-8493-2824-4